RICE UNIVERSITY

Computational Aerodynamics Modeling of Flapping Wings With Video-Tracked Locust-Wing Motion

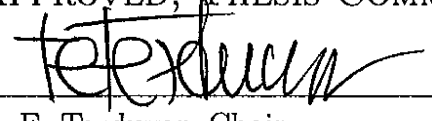
by

Anthony M. Puntel, 2nd Lt, USAF

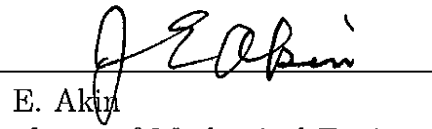
A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Master of Science

Approved, Thesis Committee:

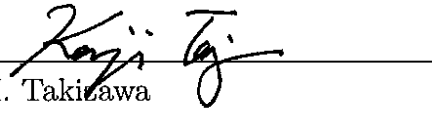


T. E. Tezduyar, Chair
Professor of Mechanical Engineering and
Materials Science



J. E. Akin
Professor of Mechanical Engineering and
Materials Science and Professor of
Computational and Applied Mathematics

A. J. Meade
Professor of Mechanical Engineering and
Materials Science



K. Takizawa
Associate Professor in Department of
Modern Mechanical Engineering and
Waseda Institute for Advanced Study
Waseda University, Tokyo, Japan

Houston, Texas
April 2012

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U. S. Government.

Abstract

Computational Aerodynamics Modeling of Flapping Wings With Video-Tracked Locust-Wing Motion

by

Anthony M. Puntel

The thesis focuses on special space–time computational techniques introduced recently for computational aerodynamics modeling of flapping wings of an actual locust. These techniques complement the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) formulation, which is the core computational technique. The DSD/SST formulation was developed for flows with moving interfaces, and the version used in the computations is "DST/SST-VMST," which is the space–time version of the residual-based variational multiscale (VMS) method. The special space–time techniques are based on using NURBS basis functions for the temporal representation of the motion of the locust wings. The motion data is extracted from the high-speed video recordings of a locust in a wind tunnel. In addition, temporal NURBS basis functions are used in representation of the motion of the volume meshes computed and also in remeshing. These ingredients provide an accurate and efficient way of dealing with the wind tunnel data and the mesh. The thesis includes a detailed study on how the spatial and temporal resolutions influence the quality of the numerical solution.

Acknowledgments

I would first like to thank Dr. Tayfun Tezduyar for the opportunity to come to Rice University and pursue research as part of the T★AFSM. The experience I have had working in this group has been invaluable, and I am truly grateful.

I would next like to thank Dr. Kenji Takizawa. This man is one of the most intelligent and hardest working people I have ever had the privilege of knowing and working with. His insight, knowledge, and efforts have proved to be priceless in the research I accomplished and knowledge I have obtained in my time at Rice. I wish him only the best he continues work at Waseda University in Tokyo, Japan and whatever the future holds for him.

I want to thank my family for alway being there for me and supporting me through all the challenges and hardships I have encountered in my life. Their love and support has brought me where I am today. To Mom and Dad, thank you for bringing me into this world, your patience as I grew, and your continued support and love every minute of the day even as we live a thousand miles apart. To my brothers, Matt and Jay, and my sister, Maddie, you have and always will be my best friends. I am constantly looking forward to the next time we can all be together and the chaos that ensues. Thank you to all of my extended family as well, although we do not see each other often, I know that you will always love and support me no matter what. Thank you all.

I also want to thank the members of T★AFSM with whom I have had the honor

of working with. To Kathleen Schjodt and Darren Montes, working with you has been amazing. Your support not just as colleagues but also friends has been an amazing pillar for me to lean on throughout the endeavors of the last two years. To Nik Kostov, thanks for your help and dedication to my research in addition to your constant words of motivation. Your hard work has proved priceless to my research. Matt Fritze, Joe Boben, and Spenser McIntyre, it was great to work and have fun with you over my time here.

To Matt Guertin and Woody Sukut, thanks for being such amazing roommates and friends over the past two years. The witty banter and ridiculous fun I have had with you will stay with me forever. To the great friends I have met and had the pleasure of spending time with over my last year here, it is too bad we could not have met sooner; the time I have spent with you will never leave my thoughts. It was short but amazingly sweet.

This work was supported by NSF Grant CRCNS-0903949. Method development and evaluation components of this work were supported also in part by ARO Grant W911NF-09-1-0346. Computational resources were provided in part by the Rice Computational Research Cluster funded by NSF Grant CNS-0821727. I would like to thank Professor Fabrizio Gabbiani and Dr. Raymond Chan (Baylor College of Medicine) for providing us with the digital data and photos extracted from the videos of the locust in their wind tunnel.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Flapping flight of insects and birds has been and continues to be an inspiration for human innovation. The prospect of joining these animals in soaring through the clouds has motivated humans to mimic these winged beasts. Some of the first flying machine designs, such as that of Leonardo da Vinci, were ornithopters. These machines were inspired by flying animals, including wings shaped similar to those of bats or birds that in turn oscillated in attempt to mimic flapping. The prospect of mimicing animal flight does not need to be limited to the prospect of human flight. The use of unmanned aerial vehicles (UAVs) and micro air vehicles (MAVs) has skyrocketed in recent years in applications to include military and civilian reconnaissance and surveillance missions.

Recent innovations in technology to include high-speed video tracking techniques and computational fluid dynamics methods have allowed for better understanding of the mechanics and aerodynamics behind flapping flight of insects and animals. This new-found understanding of flapping flight proves useful in the application of designing flapping UAVs and MAVs. Such flapping-flight-powered vehicles have the ability of accomplishing missions that larger fixed-wing or rotary-aircraft platforms cannot perform. Not only this, flapping-flight UAVs and MAVs could ultimately work

together to perform swarming missions as well.

This kind of MAV flight not only requires the understanding of the mechanics and aerodynamics of straight and level flight but also that of maneuvers, mechanics, and aerodynamics of flapping flight during collision avoidance. Collision avoidance maneuvers of flapping-flight animals requires the integration of sensory, perceptive, and muscular networks to generate the correct movement, and thus aerodynamic forces, to escape pending hazards. With this, an understanding of not only aerodynamic forces upon the wings but also the neurological and muscular responses are of interest. A combined study of animal perception, neurological, flight aerodynamics, and sensory and motor interfacing is required to truly understand what is needed for flying animal collision avoidance.

To understand the complex and integrated pieces that come together to allow these winged animals to avoid collisions, a three-pronged collaborative research effort between the Baylor College of Medicine (BCM), University of Arizona (UA) and the Team for Advanced Flow Simulation and Modeling (T★AFSM) was organized to study the aforementioned systems of flying animals. The desert locust is the study specimen. In collaboration with the BCM and UA, the T★AFSM has been using computational fluid dynamics to study the straight flight and collision avoidance aerodynamics and mechanics of the desert locust. The deformation of the locust wings in terms of both twist and camber is fairly complex and the range of motion of the locust wings over the flap cycle is quite large with a stroke amplitude (angle between top and bottom of a wingbeat cycle) of approximately 80° [20]. This complex movement and deformation of the wings calls for a computational method to deal with this elaborate moving boundary. The Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) formulation [15, 17, 18, 16, 19, 12, 13] was introduced as a general-purpose method for flows with moving interfaces. Computations using this method since its inception include free-surface and multi-fluid flows, fluid–object interaction,

flows with surfaces in fast, linear or rotational relative motion, compressible flows, shallow-water flows, fluid–particle interaction, and fluid–structure interactions (FSI), and even 3D computation of flow past a pair of flapping wings with simple motion and deformation. The T★AFSM introduced several improvements to the DSD/SST formulation. One such improvement is the introduction of an advanced turbulence model. The most recent version [12, 13] is a space–time version of the residual-based variational multiscale (VMS) method [5, 6, 3, 2] and is named "DSD/SST-VMST." Also, the use of NURBS basis functions for the temporal representation in space–time computations yields better solution accuracy in addition to providing a more accurate representation of the surface motion and deformation and a more robust and effective way of mesh moving and remeshing.

1.1 Flapping Flight

This section describes some key aspects of flapping flight. Most of the information is summarized or restated from [1] or [4]. Flapping flight is an amazing phenomenon of nature. Many beasts have wing-like appendages that allow them to glide through the air such as flying squirrels and gliding lizards. These animals produce lift using these appendages allowing them to cover fairly large distances in flight before landing. Flapping flight, on the other hand, adds a whole new factor to the mix. Although both gliding and flapping-flight animals produce lift, the fundamental difference in the outcome of flapping flight is the production of thrust. Much like a helicopter, the same lift surface provides lift and enough thrust to overcome drag, if need be. It is this ability to produce both lift and thrust from the same lifting surface that is interesting to the design of MAVs. The wing must move and deform in a way that generates forces to overcome both the drag and weight of the animal. Although the flapping motion differs between different shapes and sizes of animals, all the tip

motions, in general, follow elliptical-like patterns inclined at approximately 30° from vertical. The tip motion of the desert locust can be seen in Figure 1.1. In addition,

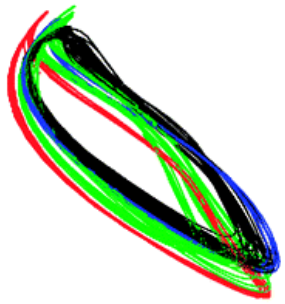


Figure 1.1: Tip path of the hindwing of a desert locust from [20].

the downstroke and upstroke of a flap cycle tends to differ. First, for almost every flying animal, the downstroke lasts longer than the upstroke. For the desert locust, the downstroke lasts nearly 90% longer than the upstroke. Second, the twist, angle of attack, and camber of the wings tend to differ between the downstroke and the upstroke. This change in wing shape is necessary to produce thrust in both the upstroke and downstroke [1].

A key factor in the production of aerodynamic forces of wings in flapping-flight animals is the presence of unsteady aerodynamic forces. Very small insects fly at fairly low Reynolds numbers, which in turn limit their lift coefficients to fairly low numbers. With the addition of unsteady forces, the lift coefficients increase to much more favorable values. One such mechanism is the clap-fling. This increases lift production by eliminating the *Wagner effect*. As a wing begins to move, before the lift-producing bound vortex builds to full strength on the leading edge of the wings, the wings must travel several chord lengths through the air. This is partially due to acceleration and partially due to the effects of the starting vortex. In flapping flight, as the wings come to the top of the stroke and begin the downstroke, these effects reduce lift production in the beginning of the downstroke. To overcome this, as the wings come to the top of the upstroke, they "clap" together. As the wings begin the downstroke

they peel away in what is called the "fling" maneuver, creating a low-pressure space between the wings. This forms two vortices with opposite rotations which establishes a bound vortex almost instantaneously. This also acts as the starting vortex for the opposite wing. This combination reduces the *Wagner effect* and in turn causes high lift production at the start of the downstroke. Another lift production mechanism includes "wake capture." In wake capture, the wing benefits from the shed vorticity of the previous stroke; a significant portion of the fluid velocity a wing encounters at the start of each stroke is due to the lingering wake of the end of the previous stroke. Lastly, "delayed stall," or "dynamic stall," is another unsteady lift producing mechanism. This comes in two forms, rotational and translational. The rotational mechanism occurs when a wing is rapidly rotated from one angle of attack to another. Here, the airflow stays attached, and the lift coefficient can spike for a short period of time. Translational stall occurs when a wing with a high angle of attack starts moving from rest. With this, the *Wagner effect* actually allows the bound vortex to grow; however, unless the angle of attack is eventually reduced, the wing will stall.

All of these unsteady mechanisms play large roles in the fluid force production of flapping wings in straight flight. As collision avoidance behaviors include abrupt changes in flight direction, speed, and orientation, we expect the unsteady and transient effects and mechanisms to have a significant role in the aerodynamics of collision avoidance as well.

1.2 Project Description

As stated earlier, the desert locust, or *Schistocerca gregaria*, is the selected specimen for this study. The UA, BCM, and T★AFSM have been working in collaboration in a three-pronged research project to understand the complex integration of the sensory mechanisms involved in detecting dangers, the motor circuitry generating flight

rhythms, and the aerodynamics of flight to generate successful collision avoidance maneuvers in desert locust. Within this, the T★AFSM deals with the modeling of locust wings, the modeling of fluid–structure interactions, and the computational fluid dynamics simulations of both locust straight flight and collision avoidance behaviors. The T★AFSM receives data to model the locust wings and flight to include high-speed camera wind tunnel data of both straight flight and collision avoidance flight of the locust.

Flapping flight of locust contains several aspects that prove to be challenging for computational methods. First, the locust contains two sets of wings forewings (FWs) and hingwings (HWs). As mentioned earlier, the range of motion of these wings is approximately 80°, and the FW and HW phase is offset creating a situation in which the wings pass by each other twice during a given flap cycle. This combination presents boundaries that move through large ranges of motion and create a "shear" effect as the wings pass each other introducing difficulties for mesh moving methods. In addition to this, the motion and deformation of the wings is largely complex, requiring the need for elaborate representation in both space and time. This thesis demonstrates techniques to account for these complex scenarios.

The T★AFSM is working on generating both a straight-flight and collision avoidance model of the locust. In this thesis, the concentration is towards a bio-inspired flapping-wing MAV using the same flapping motion and mechanics of the locust. It is necessary to obtain a functional model of straight-flight behavior before a collision avoidance model can be created. This thesis introduces a new MAV body model that differs from the original locust body model and a comparison of computations using each body. Also, several refinement studies increasing both temporal and spatial resolution are presented in this thesis to evaluate the resolution of the base case. Lastly, several different wing configuration cases are introduced to test both the beneficial and disruptive interactions between the wings as well as the role wing camber and

twist play. The wind tunnel data provided by the BCM of live desert locust in both straight flight and collision avoidance maneuvers gives the basis data needed for the T★AFSM to generate the models used in this thesis. Figure 1.2 shows pictures of a locust in the wind tunnel.

Figure 1.2: Locust flapping in wind tunnel viewed from the bottom (left) and top (right) as provided by BCM.

1.3 Overview

Chapter 2 presents the governing equations of fluid dynamics, which are the Navier–Stokes equations of incompressible flows, and the finite element formulation, which is the DSD/SST formulation with a VMS turbulence model. Chapter 3 presents special modeling techniques developed and used for this problem to include NURBS-based interpolation both temporally and spatially. Chapter 4 presents body and wing geometry modeling together with wing motion and deformation representation techniques. Chapter 5 presents the results of the base MAV computation and a comparison to the locust computation. Chapter 6 presents the results of test computations with increased temporal resolution. Chapter 7 presents the results of test computations with increased spatial resolution. Chapter 8 presents the results of several wing configura-

tion studies to test the interactions of the wings and the influence of wing twist and camber on aerodynamic forces. Chapter 9 presents conclusions that may be drawn from this study.

Chapter 2

Governing Equations and Finite Element Formulations

Flapping flight is inherently a fluid–structure interaction problem. The wings and air flowing around the wings interact and influence each other to yield the complex deformation and fluid flow that generates the forces to lift the locust into the air. As stated in Section 1.2, however, the movement and deformation of the wings is prescribed through the use of wind tunnel data provided by the BCM. With this in mind, the structure is no longer influenced by the fluid forces; therefore, the interaction is modeled with a fluid-only computation. The fluid mechanics of the flow around the flapping wings is governed by the Navier–Stokes equation of incompressible flows. These equations are presented in Section 2.1. The DSD/SST formulation including the VMS turbulence model is described in Section 2.2. New generation space–time formulations to include the use of NURBS is described in Section 2.3.

2.1 Fluid Mechanics Governing Equations

Let $\Omega_t \subset \mathbb{R}^{n_{sd}}$ be the spatial domain with boundary Γ_t at time $t \in (0, T)$. The subscript t indicates the time-dependence of the domain. The Navier–Stokes equations

of incompressible flows are written on Ω_t and $\forall t \in (0,T)$ as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{u} - \mathbf{f}\right) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \mathbf{0}, \tag{2.1}$$

$$\boldsymbol{\nabla}\cdot\mathbf{u} = 0, \tag{2.2}$$

where ρ, $\mathbf{u}$ and $\mathbf{f}$ are the density, velocity and the external force, respectively. The stress tensor $\boldsymbol{\sigma}$ is defined as $\boldsymbol{\sigma}(p,\mathbf{u}) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u})$, with $\boldsymbol{\varepsilon}(\mathbf{u}) = \left((\boldsymbol{\nabla}\mathbf{u}) + (\boldsymbol{\nabla}\mathbf{u})^T\right)/2$. Here p is the pressure, $\mathbf{I}$ is the identity tensor, $\mu = \rho\nu$ is the viscosity, ν is the kinematic viscosity, and $\boldsymbol{\varepsilon}(\mathbf{u})$ is the strain-rate tensor. The essential and natural boundary conditions for Eq. (2.1) are represented as $\mathbf{u} = \mathbf{g}$ on $(\Gamma_t)_{\mathrm{g}}$ and $\mathbf{n}\cdot\boldsymbol{\sigma} = \mathbf{h}$ on $(\Gamma_t)_{\mathrm{h}}$, where $(\Gamma_t)_{\mathrm{g}}$ and $(\Gamma_t)_{\mathrm{h}}$ are complementary subsets of the boundary Γ_t, $\mathbf{n}$ is the unit normal vector, and $\mathbf{g}$ and $\mathbf{h}$ are given functions. A divergence-free velocity field $\mathbf{u}_0(\mathbf{x})$ is specified as the initial condition.

2.2 DSD/SST-VMST (ST-VMS) Formulation of Fluid Mechanics

A space–time variational formulation of incompressible flows (see for example [15, 17, 18, 16, 19, 12]) is written over a sequence of N space–time slabs Q_n, where Q_n is the slice of the space–time domain between the time levels t_n and t_{n+1}, and P_n is the lateral boundary of Q_n. We denote the trial and test functions spaces for the velocity and pressure as $\mathbf{u} \in \mathcal{S}_{\mathbf{u}}$, $p \in \mathcal{S}_p$, $\mathbf{w} \in \mathcal{V}_{\mathbf{u}}$ and $q \in \mathcal{V}_p$. The notation $(\cdot)_n^-$ and $(\cdot)_n^+$ denotes the function values at t_n as approached from below and above. At each time step, the integrations are performed over Q_n. The essential and natural boundary conditions are enforced over $(P_n)_{\mathrm{g}}$ and $(P_n)_{\mathrm{h}}$, the complementary subsets of the lateral boundary of the space–time slab. In the DSD/SST method [15, 17, 18, 16, 19, 12], the space–time finite element interpolation functions are continuous within a space–time

slab, but discontinuous from one space–time slab to another. Each Q_n is decomposed into elements Q_n^e, where $e = 1, 2, \ldots, (n_{el})_n$. The subscript n used with n_{el} is for the general case where the number of space–time elements may change from one space–time slab to another. The finite-dimensional trial and test functions spaces are denoted as $\left(\mathcal{S}_{\mathbf{u}}^h\right)_n$, $\left(\mathcal{S}_p^h\right)_n$, $\left(\mathcal{V}_{\mathbf{u}}^h\right)_n$ and $\left(\mathcal{V}_p^h\right)_n$.

The DSD/SST-VMST method is given as

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}^h \mathbf{u}^h) - \mathbf{f}^h \right) dQ + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) dQ \\
&- \int_{(P_n)_{\mathrm{h}}} \mathbf{w}^h \cdot \mathbf{h}^h dP + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h dQ + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) d\Omega \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \left[\rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \boldsymbol{\nabla} q^h \right] \cdot \mathbf{u}' dQ - \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \boldsymbol{\nabla} \cdot \mathbf{w}^h p' dQ \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \rho \mathbf{u}' \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \cdot \mathbf{u}^h dQ - \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \rho \mathbf{u}' \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \cdot \mathbf{u}' dQ = 0,
\end{aligned} \tag{2.3}$$

where

$$\mathbf{u}' = -\frac{\tau_{\mathrm{M}}}{\rho} \mathbf{r}_{\mathrm{M}} \left(\mathbf{u}^h, p^h \right), \quad p' = -\rho \nu_{\mathrm{C}} r_{\mathrm{C}} \left(\mathbf{u}^h \right), \tag{2.4}$$

and

$$\mathbf{r}_{\mathrm{M}} (\mathbf{u}, p) = \rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} - \mathbf{f} \right) + \boldsymbol{\nabla} p - 2 \boldsymbol{\nabla} \cdot \mu \boldsymbol{\varepsilon}(\mathbf{u}), \tag{2.5}$$

$$r_{\mathrm{C}} (\mathbf{u}) = \boldsymbol{\nabla} \cdot \mathbf{u}, \tag{2.6}$$

and τ_{M} and ν_{C} are stabilization parameters closely related to $\tau_{\mathrm{SUPG}}/\tau_{\mathrm{PSPG}}$ and ν_{LSIC}.

Remark 1 *The original DSD/SST formulation was named DSD/SST-SUPS in [12] (i.e. the version with the SUPG/PSPG stabilization).*

Remark 2 *If we exclude the last two terms, the formulation becomes the same as the modified DSD/SST-SUPS formulation (where the advection term is in the conservation-*

law form) under the conditions $\tau_{PSPG} = \tau_{SUPG}$ and $\nu_{\mathrm{C}} = \nu_{LSIC}$. The 6th and 7th terms are the Streamline-Upwind/Petrov-Galerkin (SUPG) and Pressure-Stabilizing/Petrov-Galerkin (PSPG) stabilizations and LSIC (least-squares on incompressibility constraint) term.

2.3 New-Generation Space–Time Formulations

Additions to the set of new-generation space–time formulations named DSD/SST-SP, DSD/SST-TIP1, and DSD/SST-SV developed in [19], were introduced in [12], mainly the extension to the space–time formulations with NURBS, and redeployment of the original DSD/SST formulation (i.e. DSD/SST-DP) in conjunction with the VMST version. A space–time basis function can be written as a product of its spatial and temporal parts:

$$N_a^\alpha = T^\alpha(\theta)\, N_a(\boldsymbol{\xi}), a = 1, 2, \ldots, n_{ens}, \quad \alpha = 1, 2, \ldots, n_{ent}, \tag{2.7}$$

where $\theta \in [-1,\ 1]$ is the temporal element coordinate, and n_{en} and n_{ent} are the number of spatial and temporal element nodes (see Figure 2.1 for examples of temporal basis functions). In general, the values

$$\phi_n^- = \lim_{t \to t_n^-} \phi^h(t), \qquad \phi_n^+ = \lim_{t \to t_n^+} \phi^h(t), \tag{2.8}$$

do not need to be equal to $\phi_{n-1}^{n_{ent}}$ and ϕ_n^1 (coefficients of the basis functions $T_{n-1}^{n_{ent}}$ and T_n^1). However, for the cases we consider here the basis functions are interpolatory at $\theta = -1$ and $\theta = 1$, and therefore $\phi_n^- = \phi_{n-1}^{n_{ent}}$ and $\phi_n^+ = \phi_n^1$.

As pointed out in [12], different components (i.e. unknowns), and the corresponding test functions, can be discretized with different sets of temporal basis functions. This was shown in [12] by introducing a secondary mapping $\Theta_\zeta(\theta) \in [-1,\ 1]$, where

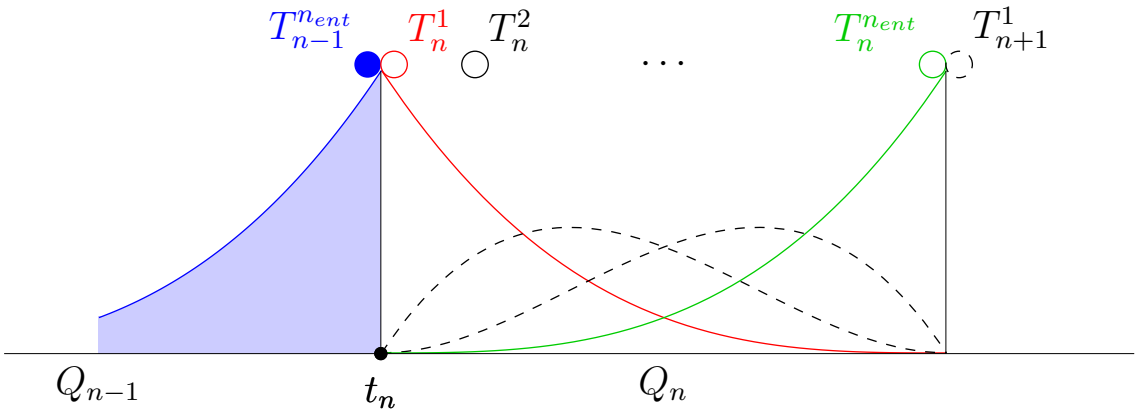


Figure 2.1: Temporal basis functions from [13].

$\Theta_\zeta(\theta)$ is a strictly-increasing function, and rewriting the generalized space–time basis function for the element indices $(a,\ \alpha)$ as

$$\left(N_a^\alpha\right)_\zeta = T^\alpha\left(\Theta_\zeta(\theta)\right) N_a\left(\boldsymbol{\xi}\right). \tag{2.9}$$

Here ζ indicates the component, which can also be "t". Prescribed and unknown variables can be represented over different space–time slabs, because we only need to supply the prescribed values at the integration points of the space–time slab used in representing the unknown variables. Most of the time higher-order basis functions can represent complex functions with fewer number of control points. This is very helpful in decreasing the I/O intensity, such as in computations with the multiscale SCFSI techniques (see Section 3 of [12]).

Chapter 3

Special Techniques

As mentioned in Section 1.3, the motion of the wings requires special mesh motion techniques as well as techniques to represent the complex motion and deformation of the wings. This chapter describes several special techniques developed by the T★AFSM to address these and other computational challenges related to modeling of flapping flight. For completeness, we provide from [9] the material in this chapter.

3.1 Time Representation

Let us represent time $t \in (0, T)$ with p^{th} order NURBS basis functions, R^β ($\beta = 0, \ \cdots, \ n_{\rm ct} - 1$). The basis functions are defined on the parametric space described by the open knot vector $\{\vartheta_1, \ \cdots, \ \vartheta_{n_{\rm kt}}\}$, where $n_{\rm ct}$ and $n_{\rm kt}$ are the number of control points and knots. Then, time t can be written as

$$t = \sum_{\beta=0}^{n_{\rm ct}-1} t_c^\beta R^\beta(\vartheta), \tag{3.1}$$

where t_c^β represents the temporal-control point. In the case of the space–time formulation there is a mesh corresponding to each temporal-control point t_c^β. With a strictly-increasing mapping function $\Theta_t(\theta)$, the element coordinate for a knot span

and the NURBS parametric space are related as follows:

$$\vartheta = \frac{(1-\Theta_t(\theta))\vartheta_{e+p+1} + (1+\Theta_t(\theta))\vartheta_{e+p+2}}{2}, \tag{3.2}$$

where e represents the element index ($e = 0, \cdots n_{elt} - 1$, where n_{elt} is the number of elements). We have $n_{\text{ct}} = n_{\text{kt}} - p - 1$, and assuming no knot multiplicity inside the knot vector, $n_{elt} = n_{\text{kt}} - 2p - 1$. We define the element shape functions as

$$T_e^{\alpha}(\Theta_t(\theta)) = R^{e+\alpha-1}(\vartheta). \tag{3.3}$$

In the time interval of element e, we represent t with the local shape functions

$$t(\Theta_t(\theta)) = \sum_{\alpha=1}^{p+1} t_c^{e+\alpha-1} T_e^{\alpha}(\Theta_t(\theta)). \tag{3.4}$$

Remark 3 *As pointed out in [9], similar to how it was in Section 2.3 with $\Theta_{\mathbf{x}}(\theta)$, the re-parametrization with the mapping function $\Theta_t(\theta)$ adds flexibility to temporal representation, which would be attractive in some cases. For example, an arc can be represented by NURBS, however we cannot represent a constant speed on the arc. The re-parametrization allows us to have a constant speed on the arc.*

3.1.1 Time Marching Problem

Consider a set of NURBS basis functions that is being used in representing the data that we will work with. Figure 3.1 shows an example. Starting from this, we can form a new basis set by knot insertion with the objective that all elements become patches as shown in Figure 3.2.

After that, we can use this basis set in our space–time computation, where the knot spans would be the time intervals of the space–time slabs, and represent the data exactly. Instead, we propose an alternative process that will have the same

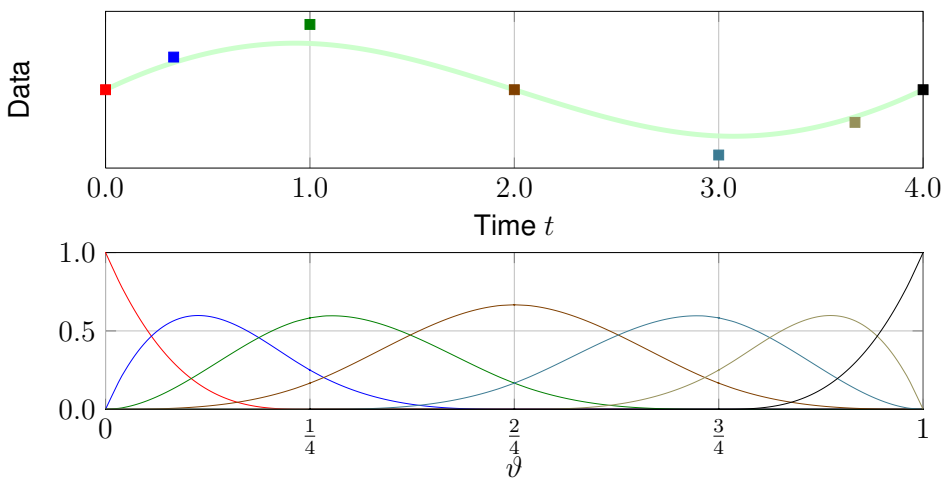


Figure 3.1: Data represented with NURBS. The data and control variables (top). The basis functions corresponding to each control variables (bottom) from [10].

functionality, but without the need to explicitly represent the data we are working with using the new basis set. In that process, we simply form a new basis set where each element is a patch, and the data is represented in our formulation in terms of its own basis set. In general, the basis set that we simply form does not need to be the same as the one that could be obtained by the process described earlier, but if it is, then the two processes result in equivalent solution methods. Figure 3.3 shows the new basis functions that we simply form.

Since we need to work with different basis sets, we map one parametric space to the other through physical time t. With the function defined by Eq. (3.1), time t can be obtained from the parametric space ϑ. Here we consider the inverse functionality; i.e., $t \rightarrow \vartheta$. We find the parametric space coordinate as follows:

1. Find the element e that is represented by the knot span $(\vartheta_{e+p+1}, \vartheta_{e+p+2})$. The process requires only time values at each element boundary, and we can quickly obtain the element index e by using a binary search technique.

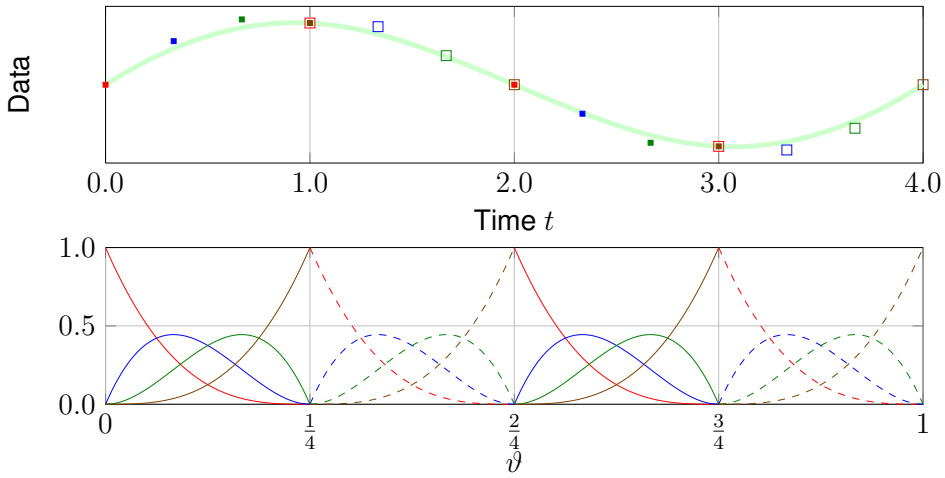


Figure 3.2: The data represented with basis functions after the knot insertion. The data and control variables (top). The basis functions corresponding to each control variables (bottom) from [10].

2. Calculate θ for a given t by using Newton–Raphson iterations as follows:

$$\theta^{i+1} = \theta^i - \left(t - t\left(\Theta_t\left(\theta^i\right)\right)\right)\left(\left.\frac{\mathrm{d}t}{\mathrm{d}\theta}\right|^i\right)^{-1} \tag{3.5}$$

where superscript "i" is the iteration counter, $t\left(\Theta_t\left(\theta^i\right)\right)$ can be calculated from Eq. (3.4), and

$$\left.\frac{\mathrm{d}t}{\mathrm{d}\theta}\right|^i = \sum_{\alpha=1}^{p+1} t_c^{e+\alpha-1} \left.\frac{\mathrm{d}T_e^\alpha}{\mathrm{d}\Theta_t}\right|_{\Theta_t(\theta^i)} \left.\frac{\mathrm{d}\Theta_t}{\mathrm{d}\theta}\right|_{\theta^i}. \tag{3.6}$$

We use as the initial guess $\theta^0 = 0$.

3. Compute ϑ from Eq. (3.2).

3.1.2 Design of Temporal NURBS Basis Functions

In the previous section we described how to find the parametric space value corresponding to physical time. Here we describe some specific temporal representations.

For implementation convenience and computational efficiency, we restrict the time interval of the space–time slab such that the time interval does not step over a time

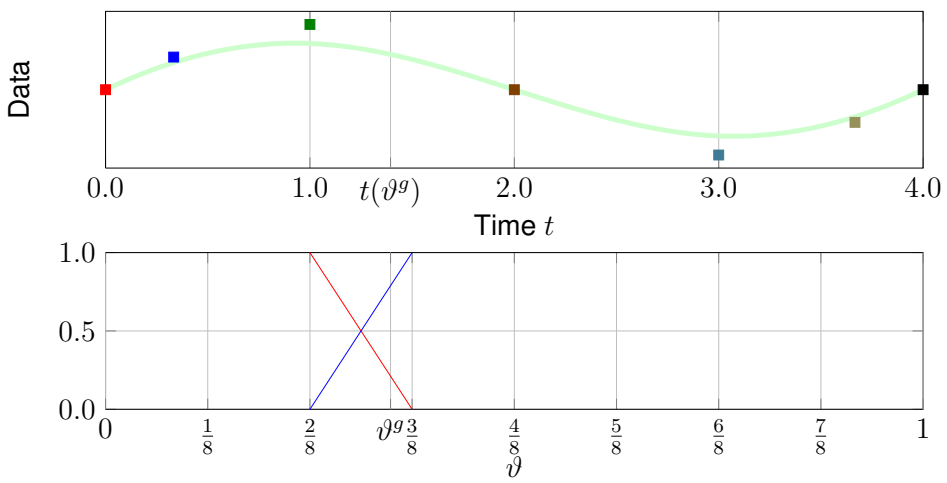


Figure 3.3: The data and control variables (top). The basis functions that we simply form for a given interval for the space–time computation (bottom). To integrate over the interval, in the NURBS representation of the data we need to search for the corresponding element and parametric coordinate for the time $t(\vartheta^g)$ of each quadrature point ϑ^g, and interpolate the value from the data from [10].

corresponding to a temporal knot in the basis set used for representing the data or mesh. Thus the supporting set of meshes for each space–time slabs consists of only specific $p+1$ meshes, where p is the order of basis used for representing the data or mesh. Because of that requirement, a uniform element size, i.e. $t(\vartheta_{e+p+2}) - t(\vartheta_{e+p+1}) = \Delta t$, where $\Delta t = \frac{T}{n_{elt}}$, is convenient. Moreover, we might have the following requirement:

$$\frac{\mathrm{d}t}{\mathrm{d}\theta} = \frac{\Delta t}{2}. \tag{3.7}$$

In the case of B-spline basis functions, for the identical mapping $\Theta_t(\theta) = \theta$, we can satisfy the condition expressed by Eq. (3.7) by selecting the control points as follows:

$$t_c^\beta = t_c^{\beta-1} + \frac{\vartheta_{\beta+p+1} - \vartheta_{\beta+1}}{p\,(\vartheta_{n_{\mathrm{kt}}} - \vartheta_1)} T, \tag{3.8}$$

for $\beta = 1, \ \cdots, \ n_{\mathrm{ct}} - 1$ and $t_c^0 = 0$.

3.1.3 Approximation in Time

Let $\boldsymbol{\chi}_A^s$ be the sampling values of a time-varying spatial position vector $\mathbf{x}_A$ at sampling times t^s ($s = 0, \cdots, n_{\text{sp}} - 1$, where n_{sp} is the number of sampling points). For example, $\mathbf{x}_A$ could be the position vector for spatial node A, or it could be the position vector for a point on a surface geometry extracted from video data. For each A, as proposed in [10, 13], we represent the path corresponding to the sampling points with NURBS. This serves two purposes: bringing smoothness to the temporal representation, and better representation accuracy for less control points. First, we form a linear finite element mesh in time, consisting of two-node elements. Then, we use least-squares projection to translate that to NURBS representation:

$$\int_0^T \boldsymbol{R}_A^h \cdot \left(\mathbf{x}_A^h - \boldsymbol{\chi}_A^h\right) \mathrm{d}t = 0, \tag{3.9}$$

where $\boldsymbol{R}_A^h$ and $\mathbf{x}_A^h$ are the test function and NURBS representation in time, and $\boldsymbol{\chi}_A^h$ is the linear representation in time. Thus, we obtain the control point $\mathbf{x}_A^\beta$ corresponding to each time control point t_c^β. Figure 3.4 is an example.

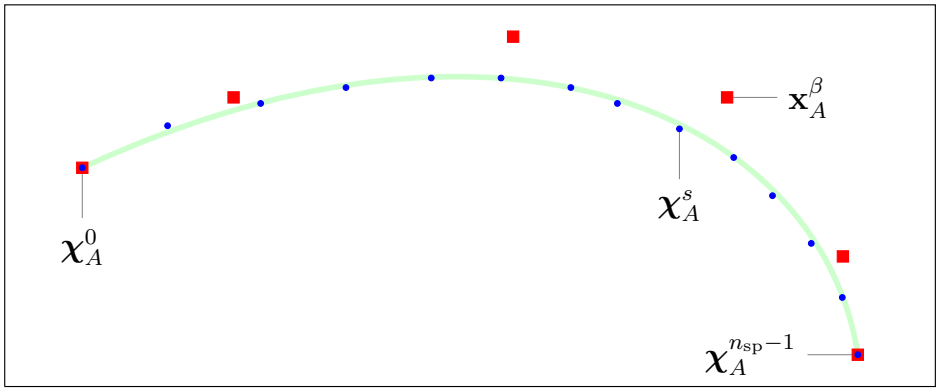


Figure 3.4: NURBS representation for a time-varying spatial position vector. The circles are the spatial position vector at each sampling time. The squares are the temporal-control points and the smooth curve is represented by them from [10].

Remark 4 *This is a simple projection. However, the concept is applicable to more*

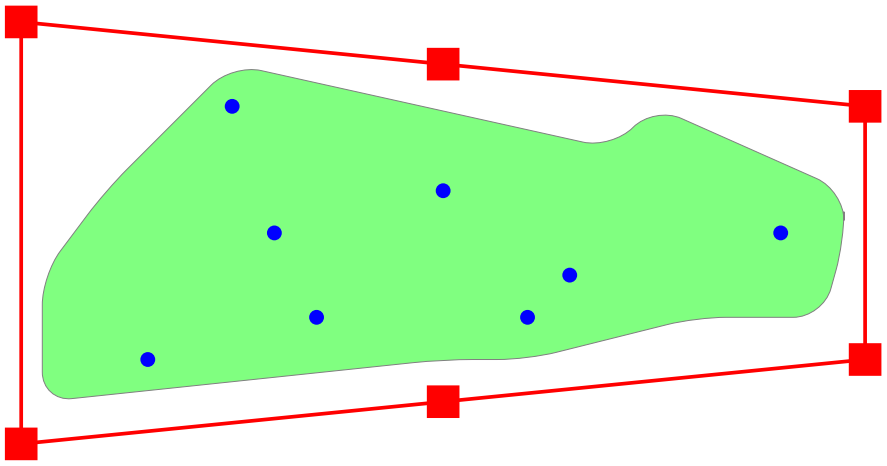


Figure 3.5: SSDM. Complex shape is shaded. Circles are tracked points. SS is represented by squares (control points) from [10].

complicated formulations to obtain smoother motion.

3.2 Simple-Shape Deformation Model (SSDM)

Suppose we want to track the motion/deformation of an object with surface shape that is too complex to track in full detail, giving us just the option of tracking only a finite number of points belonging to this complex shape. In the simple-shape deformation model (SSDM), we assume that those tracked points are associated with a simple shape (SS) instead of the actual, complex shape. NURBS is used for the spatial representation of the SS. We note that the SS is larger than the complex shape. The complex shape can be represented by finite elements or NURBS.

Starting with the reference configuration, the SS, the complex shape and the tracked points all can be seen in a common parametric space (Figure 3.5). The complex shape can be represented by finite elements or NURBS. Control points of the SS at different times during tracking are determined by a least-squares fit. The fit minimizes the difference between the positions on the SS (with respect to the reference configuration) of the tracked points and the positions of the actual tracked points. The

complex shape at a given temporal-control point is determined by interpolation from the parametric space in the case of the finite element representation, and by least-square projection in the case of NURBS representation. The least-squares integration is over the parametric space of the complex shape, and we minimize, with respect to the control points of the complex shape, the difference between the complex-shape and simple-shape representations.

In the full space–time representation, the method we described above is applied to temporal-control values that are determined as described in Section 3.1.3, instead of the actual physical locations.

3.3 Mesh Update Techniques in the Temporal NURBS Representation

3.3.1 Mesh Computation and Representation

Given the surface mesh, we compute the volume mesh using the mesh moving technique we have been using [14]. Here, as proposed in [10], we apply this technique to computing the meshes that will serve as temporal-control points. This allows us to do mesh computations with longer time in between, but get the mesh-related information, such as the coordinates and their time derivatives, from the temporal representation whenever we need. Obviously this also reduces the storage amount and access associated with the meshes. However, because of the longer time between the control meshes, linear interpolation of the surfaces between control points in time might be needed in computing those meshes with the mesh moving technique mentioned.

Remark 5 *We note that getting the meshes used in the computations from the temporal representation can be done independent of which time direction was used in*

computing the control meshes.

3.3.2 Remeshing Techniques

In many computations remeshing becomes unavoidable. Two choices were proposed in [10]. To explain those two choices, let us assume that when we try to move from control mesh M_c^β to $M_c^{\beta+1}$, we find the quality of $M_c^{\beta+1}$ to be less than desirable. In the first choice, which is called "trimming," we remesh going back to $M_c^{\beta-p+1}$. Then whenever our solution process needs a mesh, depending on the time, we use the control meshes belonging to either only the un-remeshed set or only the remeshed set (Figure 3.6).

In the second choice, we perform knot insertion p times in the temporal representation of the surface at the right-most knot before the maximum value of the basis function corresponding to $t_c^{\beta+1}$, making that knot a new patch boundary. Then we do the mesh moving computation for the control meshes associated with the newly-defined basis functions, not only the one at the new patch boundary, but also going back $(p-1)$ basis functions (Figure 3.7).

We use the second choice in computations, because we believe that in many cases the need for remeshing is generated by a topological change, which we can avoid going over with a large step if we use the knot insertion process.

3.4 Fluid Mechanics Computation with Temporal NURBS Mesh

3.4.1 No-Slip Condition on a Prescribed Boundary

Suppose we have a prescribed mesh motion, and no-slip conditions on part of the boundary of that mesh. Those Dirichlet conditions can be obtained from the mesh

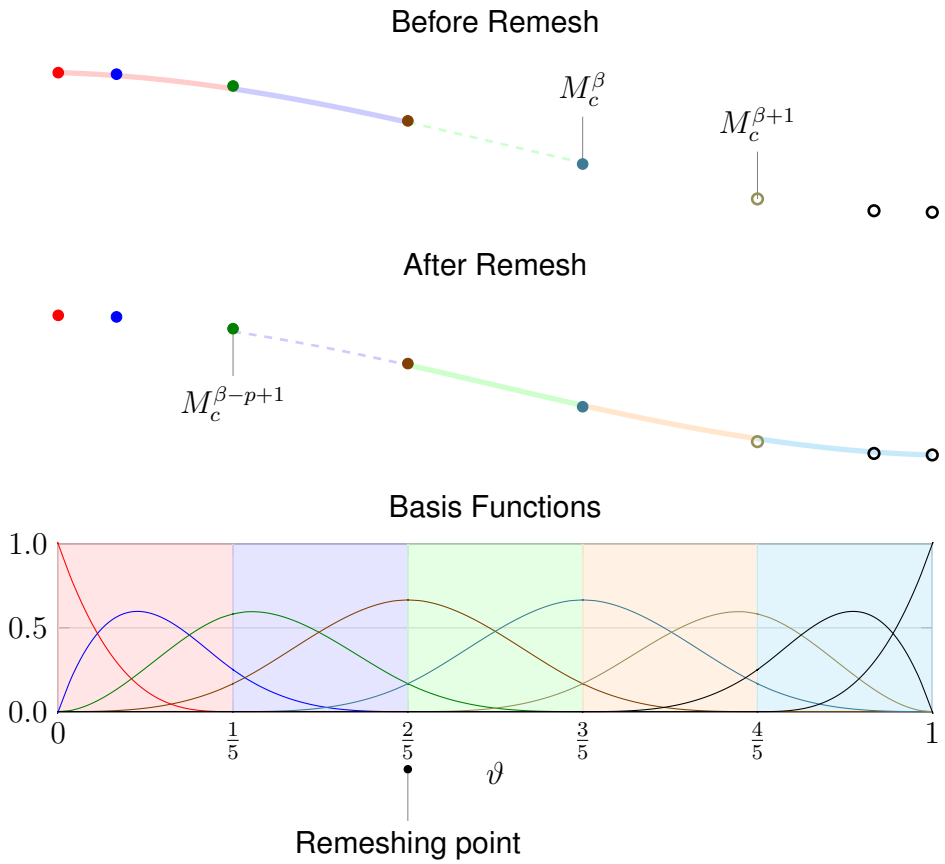


Figure 3.6: Remeshing and trimming NURBS. A set of un-remeshed meshes (top). A set of remeshed meshes (middle). Common basis functions (bottom) from [10].

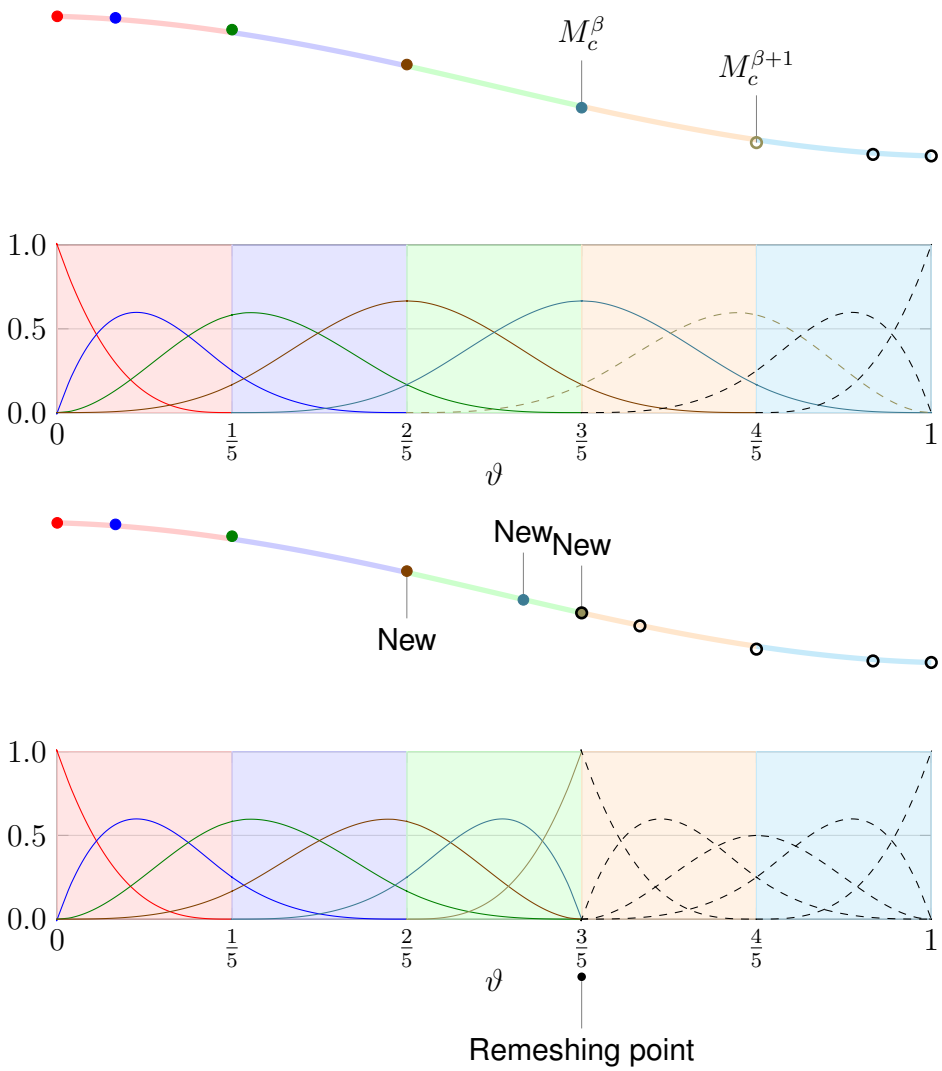


Figure 3.7: Remeshing with knot insertion. For the set of un-remeshed meshes, there are p newly-defined basis functions and the corresponding control points are marked "New." We carry out the mesh moving computations for those meshes from [10].

boundary motion.

Prior to solving the equations using a space–time slab Q_n, we use a least-squares projection for each prescribed node A as follows:

$$\int_{t_n}^{t_{n+1}} \boldsymbol{R}_A^h \cdot \left(\mathbf{u}_A^h - \frac{\mathrm{d}\mathbf{x}_A^h}{\mathrm{d}t} \right) \mathrm{d}t = 0, \tag{3.10}$$

where $\boldsymbol{R}_A^h$ represents the test function, $\mathbf{u}_A^h$ is represented by temporal-control velocities (unknown) and the corresponding basis functions in time, and the mesh velocity is obtained by the derivative of the mesh displacement, which is also represented by temporal-control positions and their basis functions. We note that $\mathbf{u}_A^h$ at time t_n approaching from below and above might be different.

Chapter 4

Flapping-Motion and Geometry Representation

This chapter describes from [9, 11] the techniques used to model the body and wings of the locust and MAV in addition to the techinques used to represent the motion and deformation of the flapping wings.

4.1 Wing and Body Shape

Based on a digital image of a pair of locust FW and HW, we construct the surface geometries of the FW and HW using a NURBS-based design software (see Figure 4.1). The FW and HW are modeled as a single NURBS patch with degeneration at the wingtip. There are 21 control points for the FW, and 51 for the HW (see Figure 4.2). This is slightly different than those used in the locust computations. In addition, the HWs for the MAV are attached to the body through the entire chord length while the HWs for the locust are not (see Figure 4.3).

The locust body dimensions are based on empirical height and width measurements provided at five cross-sectional positions. We estimate the axial curvature of the body from video images of the locust flying. The MAV body is a geometrically

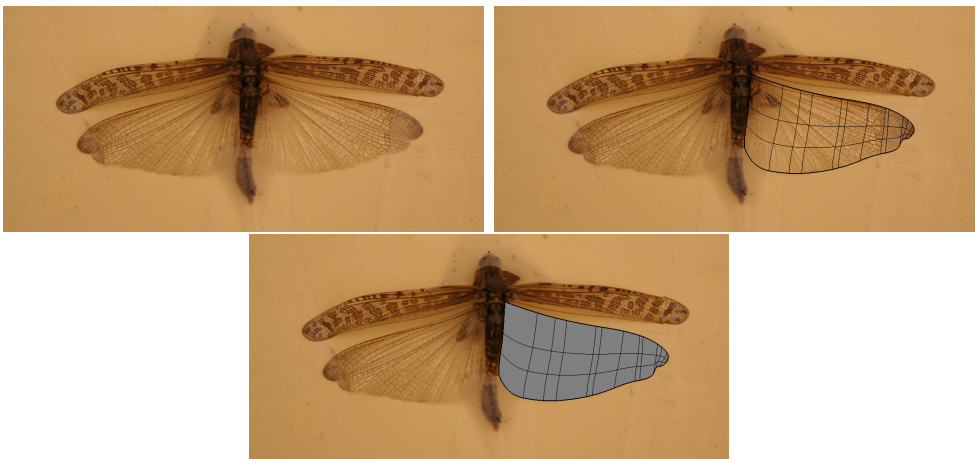


Figure 4.1: Wing model construction.

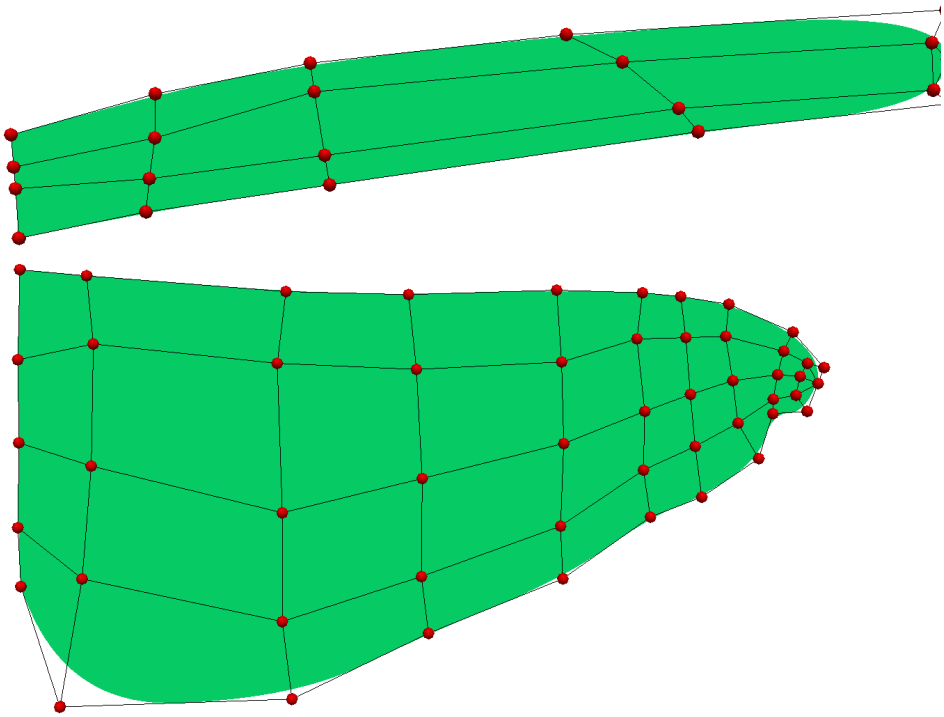


Figure 4.2: MAV FW and HW surfaces represented by NURBS and the corresponding control mesh.

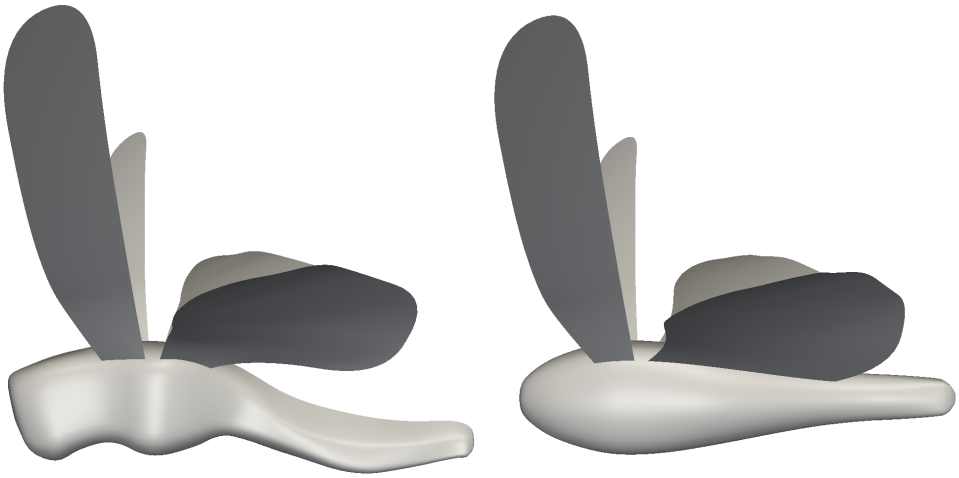


Figure 4.3: Locust body and wings, see [9], (left) and MAV body and wings (right).

simplified version of the locust body and is inspired by UAV designs currently under use. Both bodies are symmetric along the sagittal plane. The bodies are made up of 20 NURBS patches, 10 patches in each sagittal half (this is the minimum number needed to attach the wings to the body). We shape the patches as shown in Figure 4.4. The wing spatial-control meshes are deformed to the various positions in the

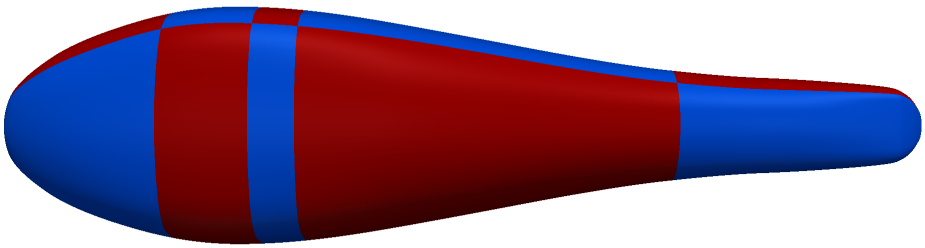


Figure 4.4: MAV body patches.

flapping motion as we will describe in Section 4.2.

4.2 Flapping-Motion Representation

The wing motions we prescribe need to accurately represent the wing motion and deformation patterns during flight. Here we face the challenge of reconciling data acquired through photogrammetry with data that is suitable as an input for computational analysis. In this computation, we use a data set that is more recent than the one reported in [10]. Both data sets were provided by our collaborators at BCM. The location of the tracking points provided in this data set can be seen in Figure 4.5. By averaging the values corresponding to each other across the symmetry plane, we create a new data set of tracking points for one side. With the help of these

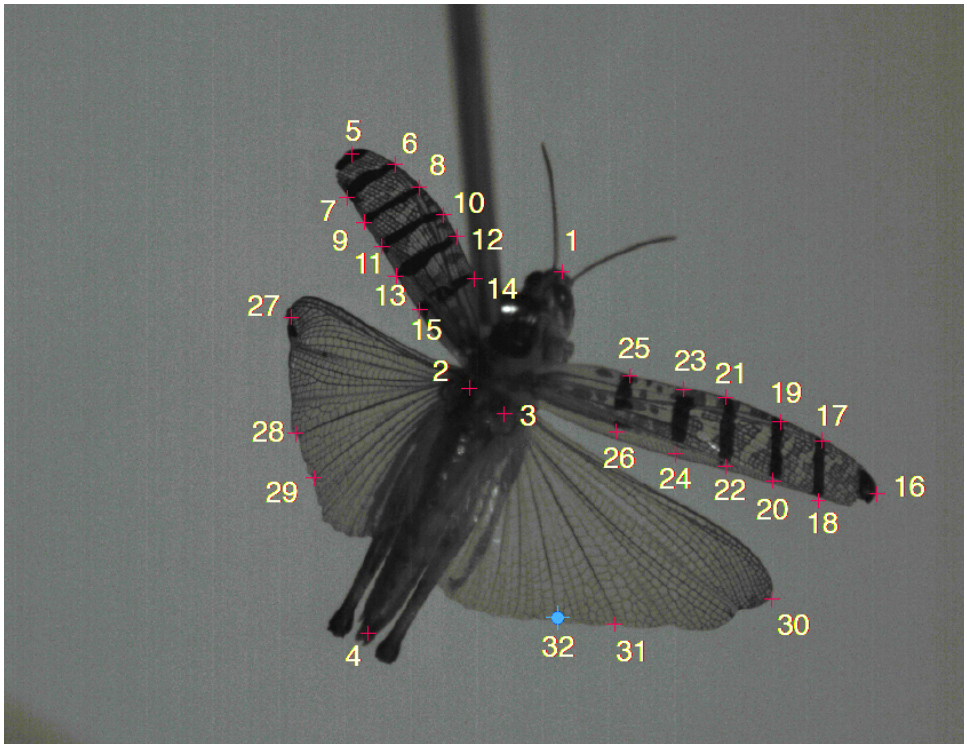


Figure 4.5: Tracking points in data set provided by BCM.

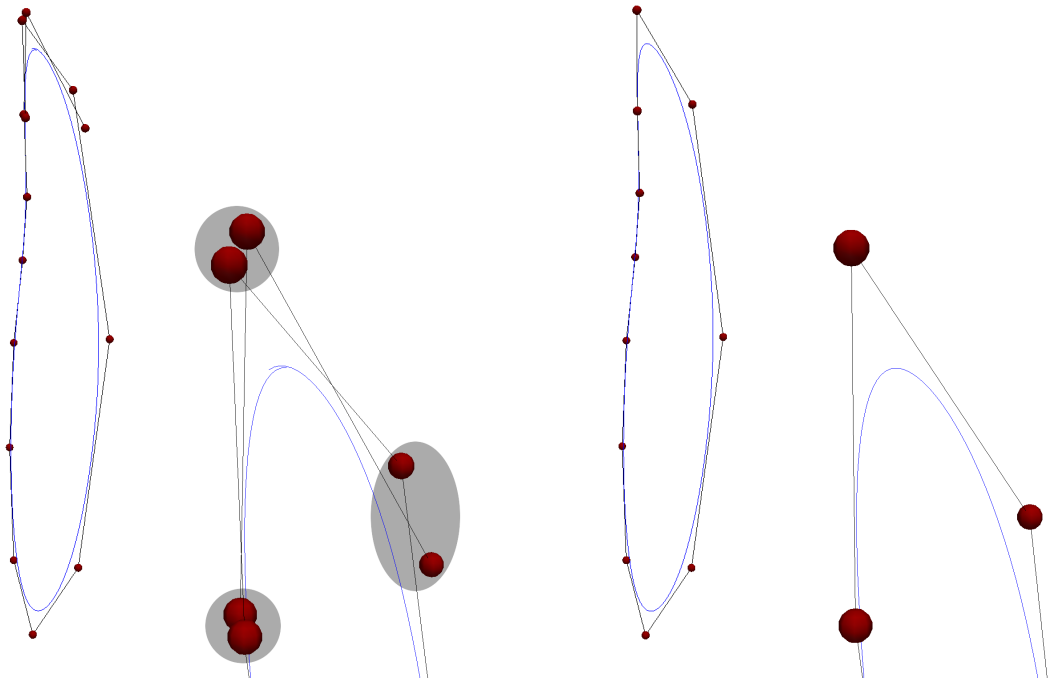
tracking points, we generate additional "tracking points" to more closely represent locust flight characteristics observed in the video. The motion is reflected across the sagittal plane of the locust to create symmetric flapping motion. The total number of tracking points used in the computations is 68. We then use this symmetric data

as the input for the least-squares projection used for the temporal representation. Next, we temporally interpolate our representative data set. For each tracking point, we apply a temporal NURBS representation as discussed in Section 3.1.2 and 3.1.3. Higher-order basis functions should be used in temporal interpolation to achieve a continuous acceleration. We use cubic B-spline functions for temporal interpolation. Motion with cubic basis functions maintains C^2-continuity across knots. With this, acceleration will be continuous across temporal knots.

4.2.1 Generating a Periodic Data Set

While flapping motion kinematics is inherently cyclical, it is not necessarily periodic. For example, the vertical position at the start of the first downstroke does not correspond to exactly the same position for subsequent strokes. This adds to the difficulty of computing flapping-flight aerodynamics. It is beneficial to have a periodic data set for a single flap cycle, where the first and last points of the cycle are colocated and have the desired continuity. Thus, a single set of deformed meshes can be appended to produce as many flapping cycles as are required. Because of the periodicity of the motion, we can compute a desire number of flapping cycles that we think would be sufficient to reach a solution that we would consider periodic.

To obtain a periodic data set, after the least-squares projection, we extract one flapping cycle by averaging and inserting knots at the top of the HW cycle (end of the upstroke and beginning of the downstroke). To maintain continuity, the control points corresponding to the knot at the beginning of the flap cycle are colocated with the control points corresponding to the knot at the end of the flap cycle (for a cubic B-spline, three control points correspond to a given knot). To obtain such repetition we average those control points. Finally, we insert knots to extract a single cycle. We show the averaging process for a cubic B-spline in Figure 4.6.



(a) Flap cycle of interest before averaging control points (left) and enlarged view of top of cycle (right). The control points we average are highlighted.

(b) Flap cycle of interest after averaging control points (left) and enlarged view of top of cycle (right).

Figure 4.6: Process used for building a periodic flapping cycle.

4.2.2 Motion of the Wings

We spatially represent the tracking points at each temporal-control point. Spatial interpolation is accomplished using the SSDM described in Section 3.2. The FW and HW SS consist of 6 and 9 control points, respectively. The mapping between the tracking points and the SS and the mapping between the SS and the wing surface for the FW and HW can be seen in Figure 4.7. We specify the control points of the

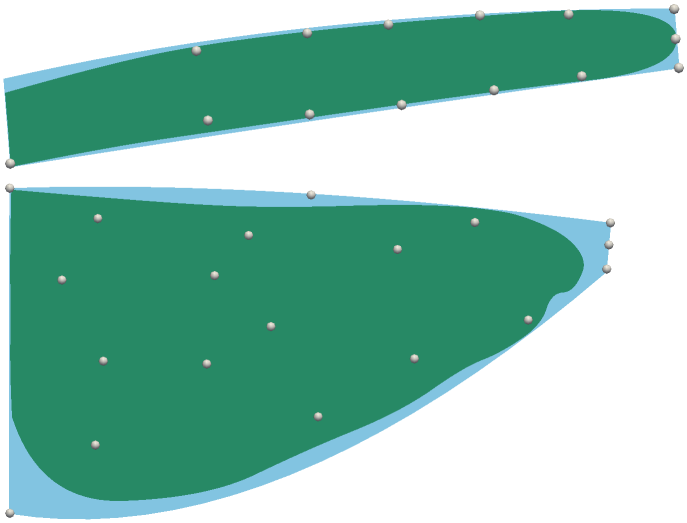


Figure 4.7: Tracking points, SS, and wing surface configuration used for mapping.

SS near the wingtip as prescribed conditions for the least-squares fit of the tracking points and the SS. This is in addition to prescribing the control points of the SS near the root chord. This technique provides a better representation of the SS motion. To find the shape of the complex-wing surface, a least-squares projection is used to fit the displacements of the SS onto the complex-wing surface. This is also done at each temporal control point. A comparison of the tracking points, SS, and complex-wing-shape at one temporal control point can be see in Figure 4.8. With the complex-wing surface projection accomplished, the wing motion is represented both spatially and temporally with NURBS.

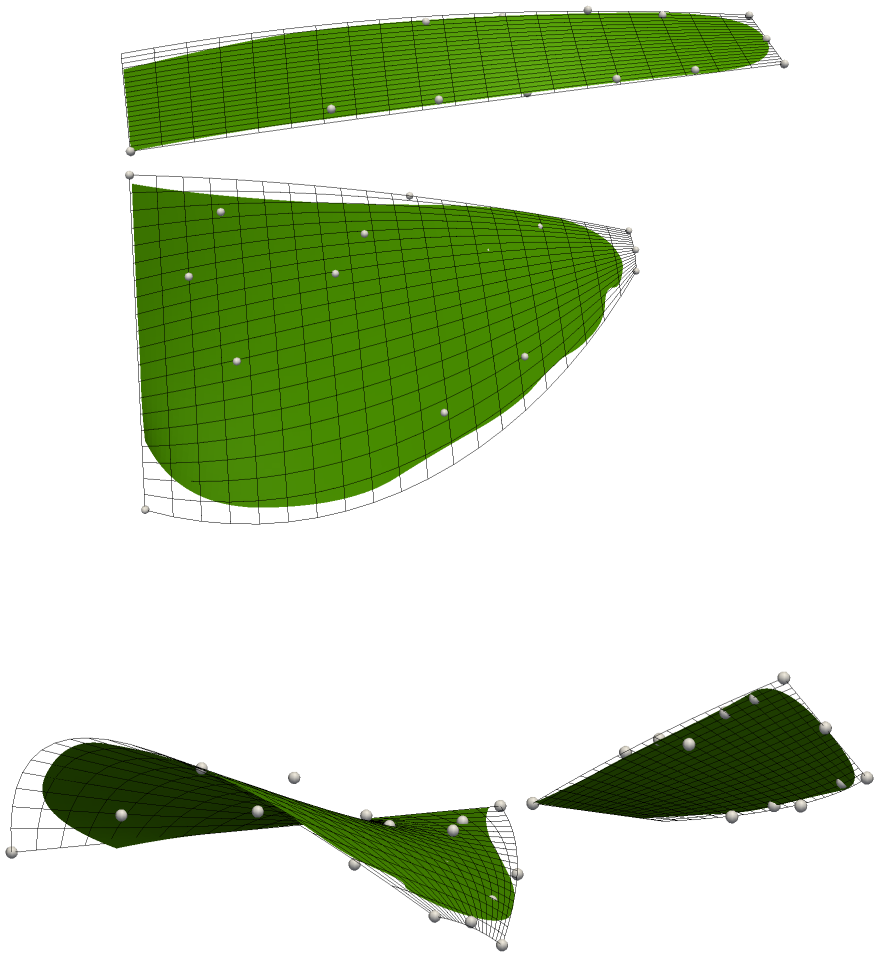


Figure 4.8: Tracking points, SS, and wing surface at one temporal control point viewed from the top (top) and side (bottom).

Chapter 5

Base MAV Computation

The motion and deformation patterns of the wings are prescribed as described in Chapter 4. We use the same wing motion as that used for earlier locust computations for the analysis of the bio-inspired flapping-wing aerodynamics of an MAV. In this chapter we describe from [11] the setup for the computation of the base MAV case, mainly the surface and volume meshes and the mesh update technique, and the results as compared to the most recent locust data from [9].

5.1 Surface and Volume Meshes

After the wing spatial-control surfaces are deformed to the various positions in the flapping motion as described in Section 4.2.2, we discretize the surfaces at each temporal-control point with triangular elements. The triangular surface mesh used in the computation is shown in Figure 5.1. We note that both wings have a finite thickness of 1% of the FW root chord, which tapers to zero thickness at the wing edges.

We generate the tetrahedral element volume mesh with the following steps. First, we generate a one-layer refinement region near the wing surface. The element height of this layer is 10% of the FW root chord. In addition, we have a cylindrical region

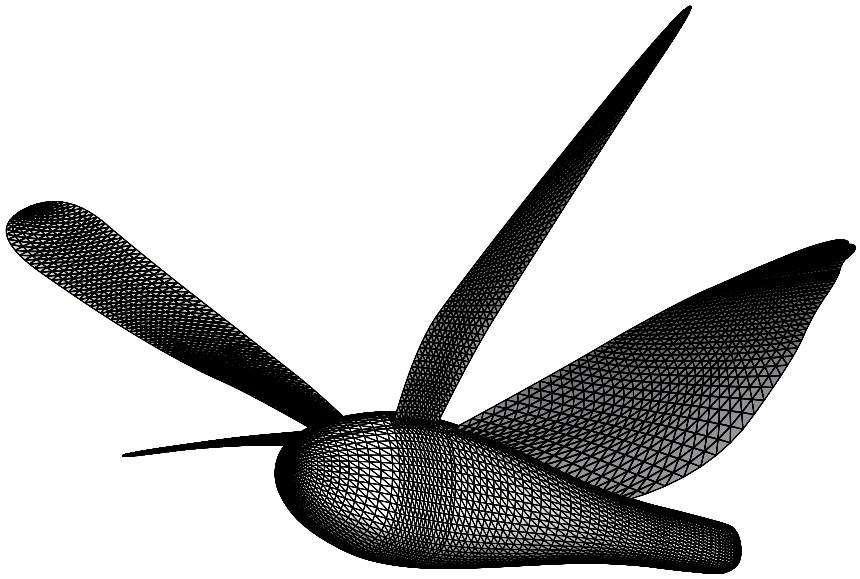


Figure 5.1: Wing and body surface meshes with triangular elements.

of increased element refinement around the locust. We also specify increased volume-mesh refinement in the region between the FW and HW. This refinement reduces the mesh distortion due to the staggered motion of FW and HW. The volume mesh within this cylindrical region is then generated using an automatic mesh generator. We rotate the cylindrical region to an angle representing the approximate free-flight body angle of the locust and compute mesh motion only in this cylindrical region as discussed in Section 4.2. A volume mesh between the cylindrical region and the external domain boundaries is then generated with an automatic mesh generator. This region is fixed for the entire computation; therefore, it is used for all temporal patches. The volume mesh boundaries and cylindrical refinement region are shown in Figure 5.2.

5.2 Mesh Update Technique

Because of the topological changes caused by the two-wing motion, we cannot use the same connectivity during the entire computation. We know the wing motion in

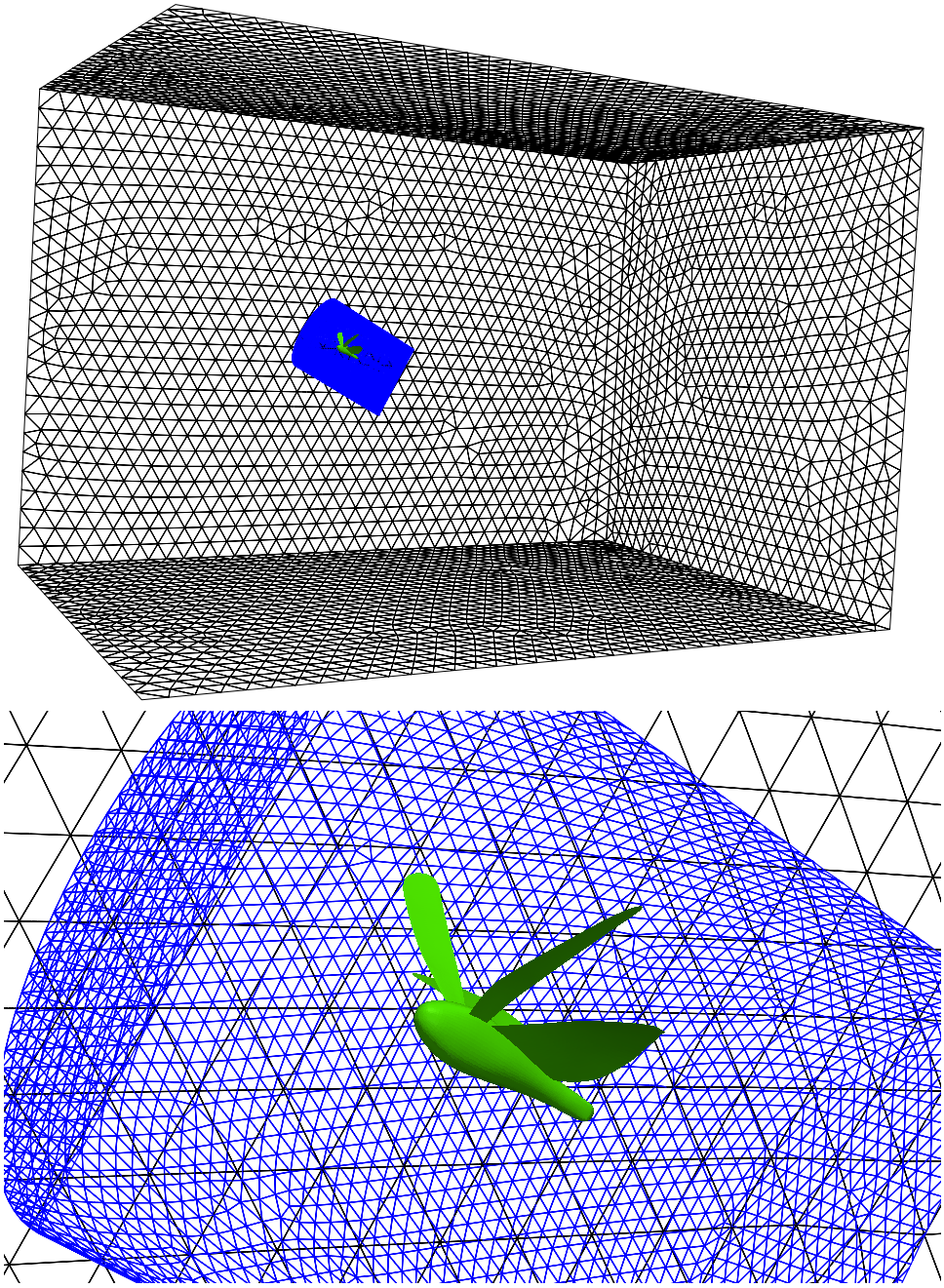


Figure 5.2: Surface meshes on some of the external computational boundaries and on the outer surface of the inner, cylindrical mesh region, which has been rotated to account for in-flight body angle.

advance, so we use a "prepared" remeshing technique.

We have a surface modeling using a temporal NURBS representation. First we decide the remeshing instants. Then, we use knot insertion to split the flapping cycle into patches as seen in Section 3.3.2. Those patches are called "temporal patches." The number of nodes and elements in the total volume mesh varies between each temporal patch. The average number of nodes and elements are 0.35 million and 2.1 million (see Table 5.1). Due to the relatively large change in deformation between each

Temporal Patch	Control Points	Knot Spans	
1	7	4	
2	6	3	
3	5	2	
4	5	2	
Temporal Patch	**Meshing Point**	**Number of Nodes**	**Number of Elements**
1	4	347,312	2,072,463
2	3	342,501	2,043,814
3	3	334,840	1,998,235
4	3	385,900	2,303,385

Table 5.1: Summary of computational setup. In each temporal patch we indicate the number of temporal control points and at which point we generate the tetrahedral volume mesh with the number of nodes and elements shown.

temporal-control point, we use subiterations for the mesh computation to divide the steps between temporal-control points into 20 smaller steps. We move the mesh, which corresponds to the middle control point, backward and forward through each smaller step using 1,500 GMRES iterations (this is the technique described in Section 3.3.1).

5.3 Computation of the Base Case

5.3.1 Computational Conditions

Figure 5.3 shows the length scales involved in the model used in the computations.

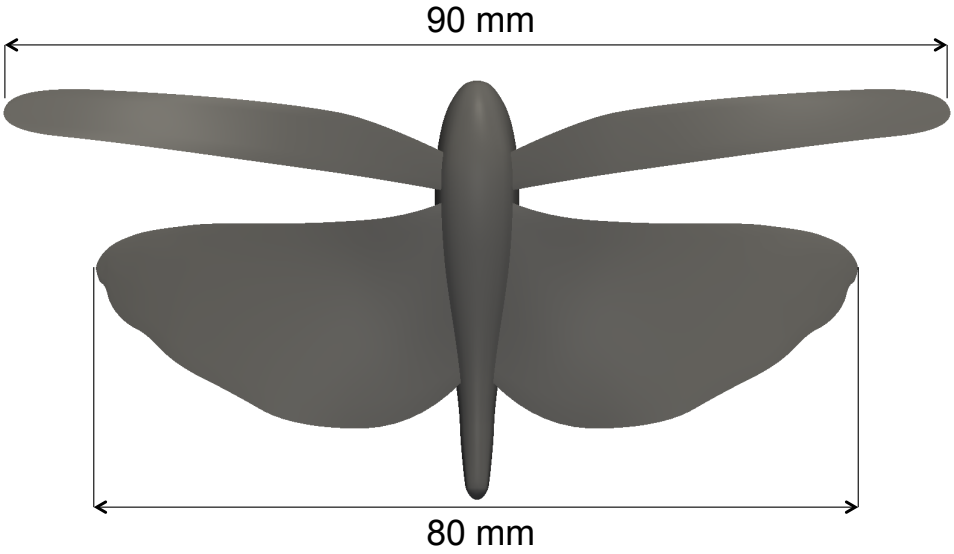


Figure 5.3: Length scales involved in the model used in the computations.

The air density and kinematic viscosity are 1.2251 kg/m^3 and 1.4606×10^{-5} m^2/s. The period of flapping is $T = 0.047$ s. Prior to the beginning of the prescribed flapping motion, we compute 400 time steps to develop the flow field. Over the first 300 time steps of this computation, we use a Cosine form to smoothly increase the inflow velocity from 0.0 to 2.4 m/s, which represents the average wind tunnel velocity used for the empirical data collection. For the last 100 time steps we keep the velocity steady at 2.4 m/s. In this flow-development computation, the time step size is 1.71×10^{-4} s, with 4 nonlinear iterations per time step. We use the DSD/SST-SUPS and DSD/SST-VMST techniques (see [12, 13] for the terminology) for the first two and last two nonlinear iterations, respectively, with the stabilization parameters as given by Eqs. (7)–(11) in [19] for $\tau_{\mathrm{M}} = \tau_{\mathrm{SUPG}}$ and Eqs. (35)–(37) in [10] for ν_{C}. The stabilization parameters are calculated after the predictor step and after the first two nonlinear iterations. The number of GMRES iterations for the nonlinear iterations are 30, 60, 120, and 180.

For the computation with flapping, we use 25 space–time slabs (with linear basis functions) for each of the knot spans in the temporal representation of the mesh,

and that gives us the time-step size of 1.71×10^{-4} s. The rest of the computational parameters are the same as those above.

Remark 6 *We have seldom observed close-to-zero or negative diagonal terms when the time-step size is large. This occurs when the fine-scale velocity is much larger than the coarse-scale (discrete) velocity. We believe this is due to the predictor, which assumes all velocities are the same from the previous time step except those with Dirichlet conditions.*

We partition this mesh in the same way as done in the previous computation to enhance parallel efficiency. Mesh partitioning is based on the METIS [8] algorithm. We partition it into 128 parts and use 128 CPU cores.

5.4 Results

We compute the problem for three complete flapping cycles. We display results for the third cycle. The lift and thrust (pressure component only) are shown in Figures 5.4–5.6 for the locust, data from [9], and MAV. Figures 5.7–5.8 show the pressure on the wing surfaces (relative to the free-stream pressure) at different instants during the third flapping cycle. Figure 5.9 shows the vorticity magnitude at different instants.

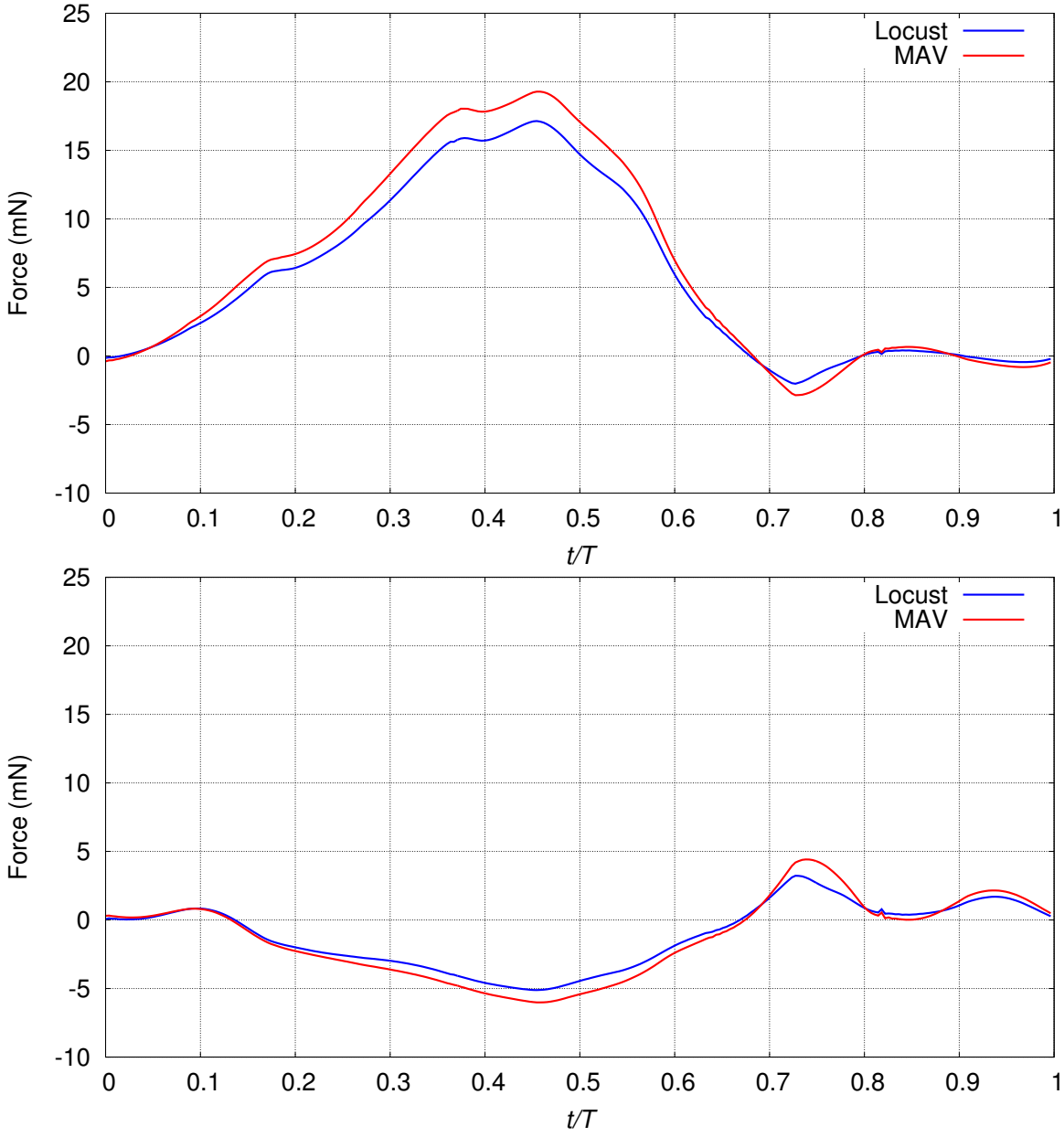


Figure 5.4: Total lift (top) and thrust (bottom) generated over one cycle. Positive value indicates that thrust exceeds drag.

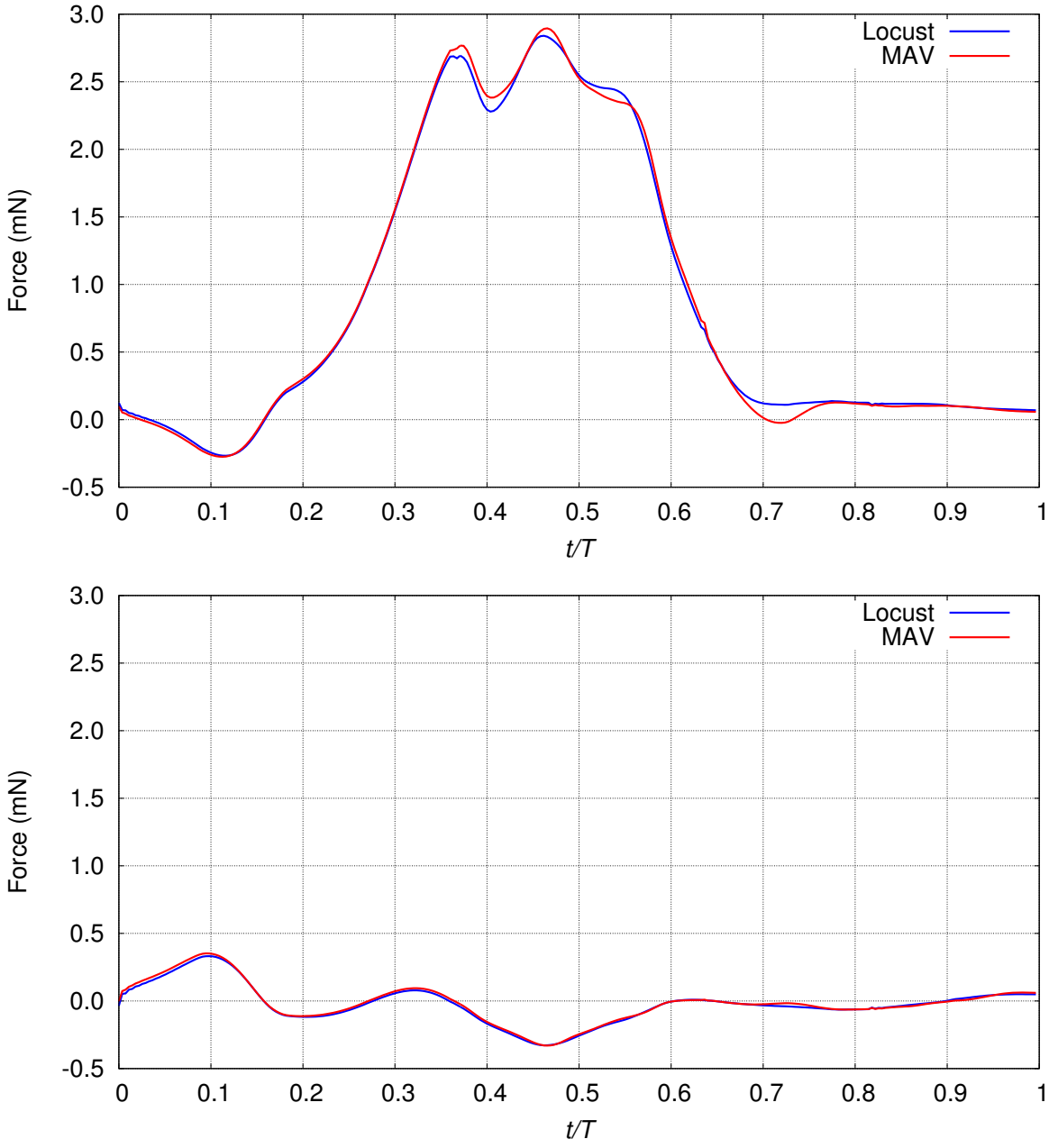


Figure 5.5: Lift (top) and thrust (bottom) generated on the right FW.

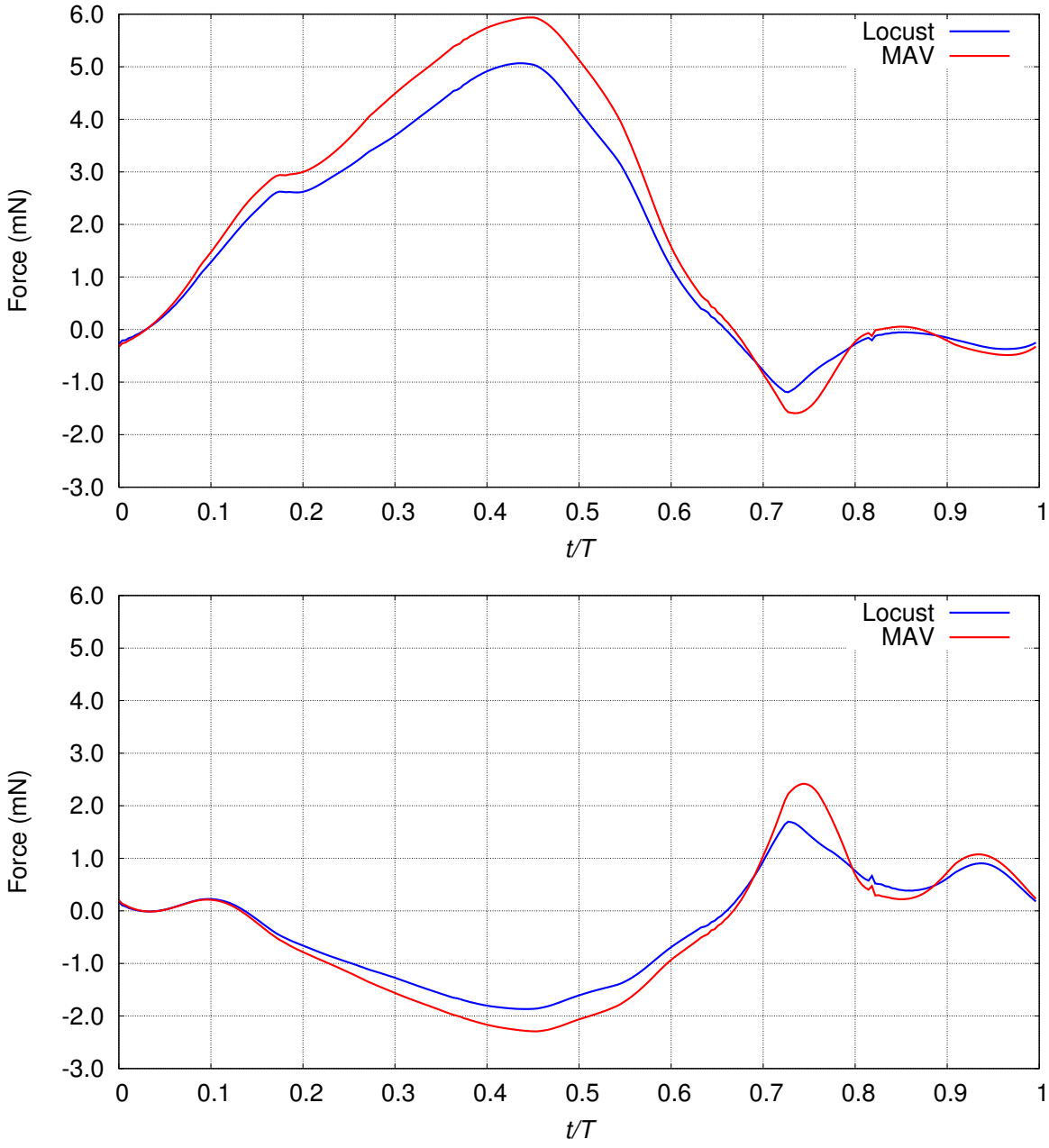


Figure 5.6: Lift (top) and thrust (bottom) generated on the right HW.

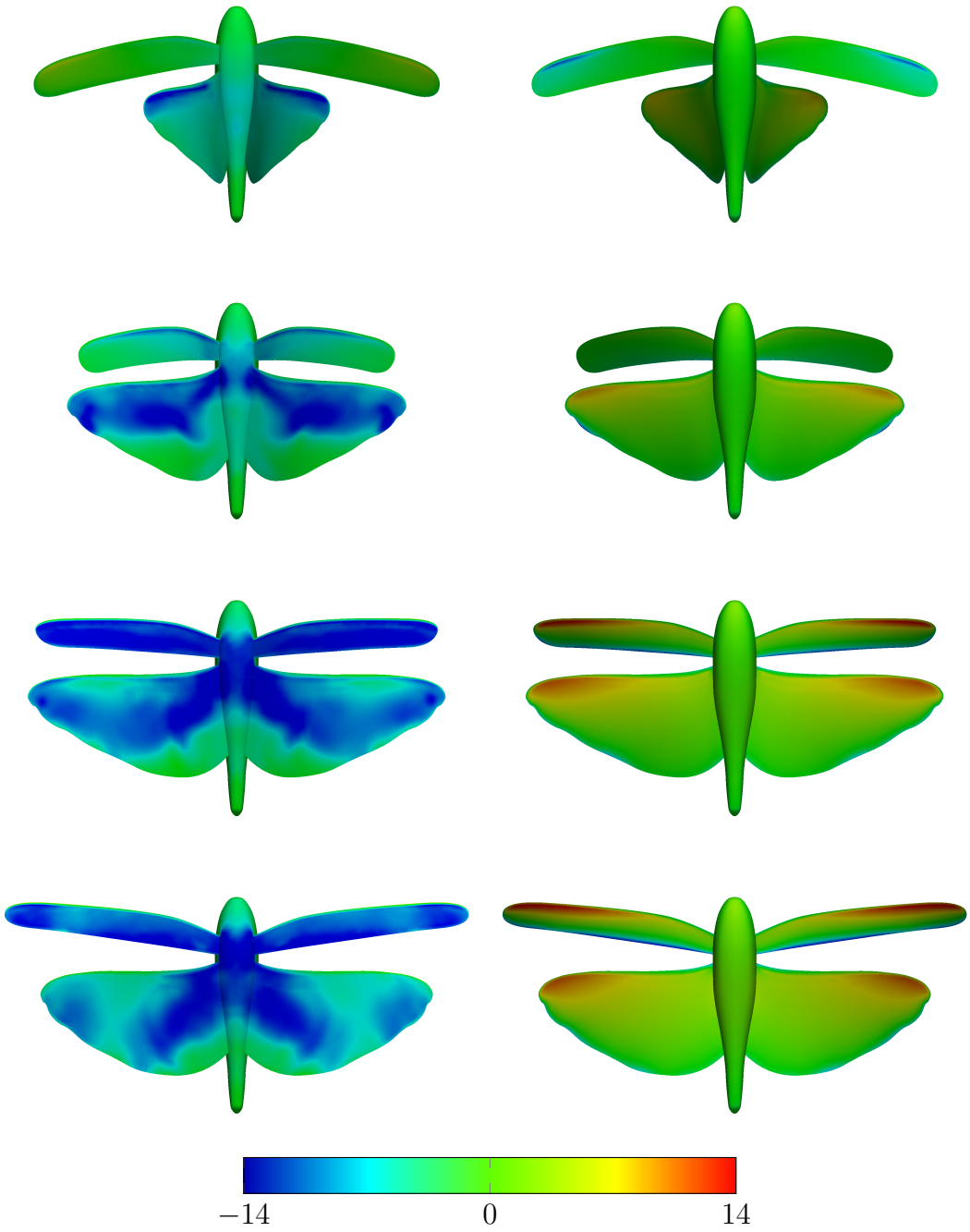


Figure 5.7: Surface pressure in Pa (relative to the free-stream pressure) at the first four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right).

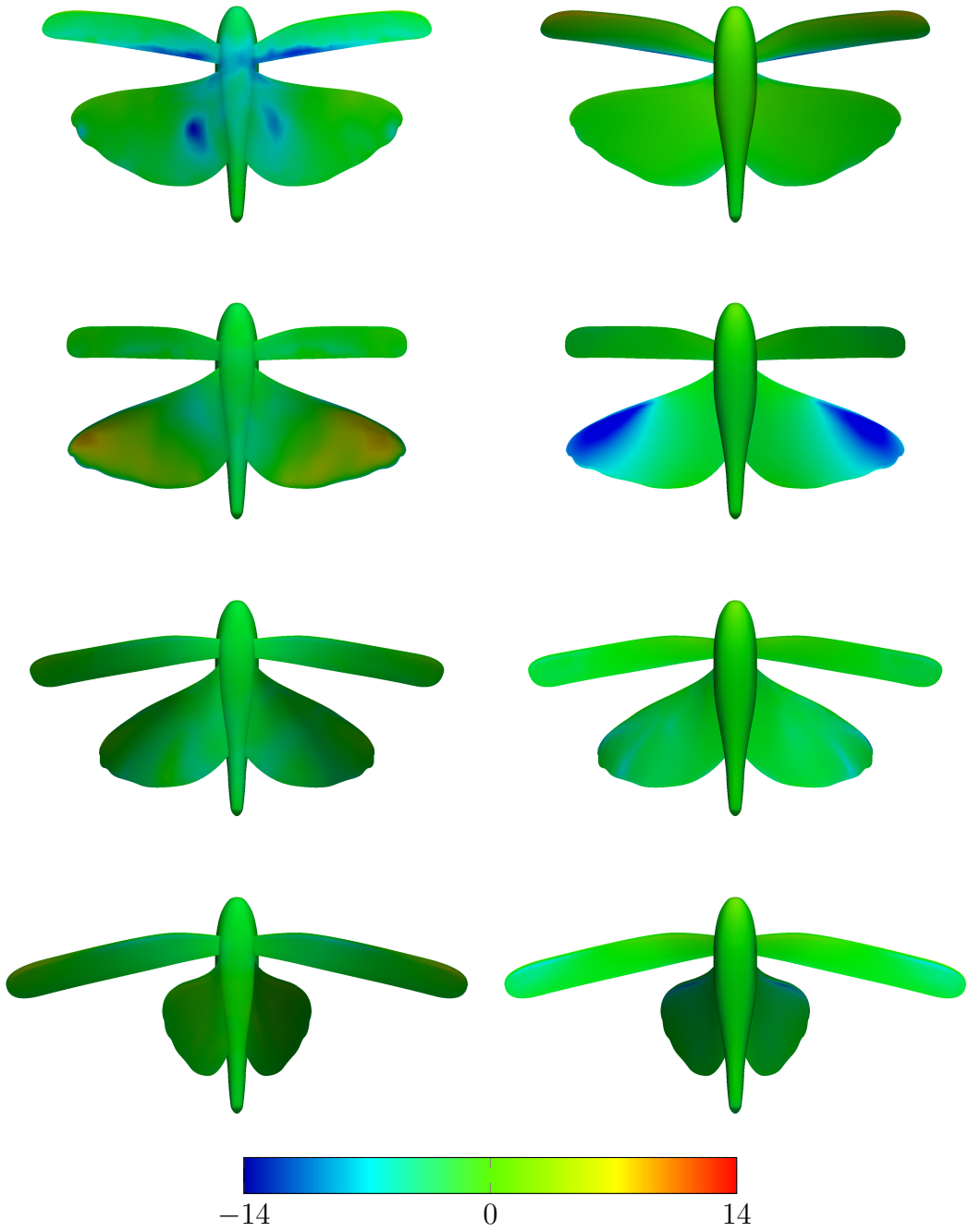


Figure 5.8: Surface pressure in Pa (relative to the free-stream pressure) at the last four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right).

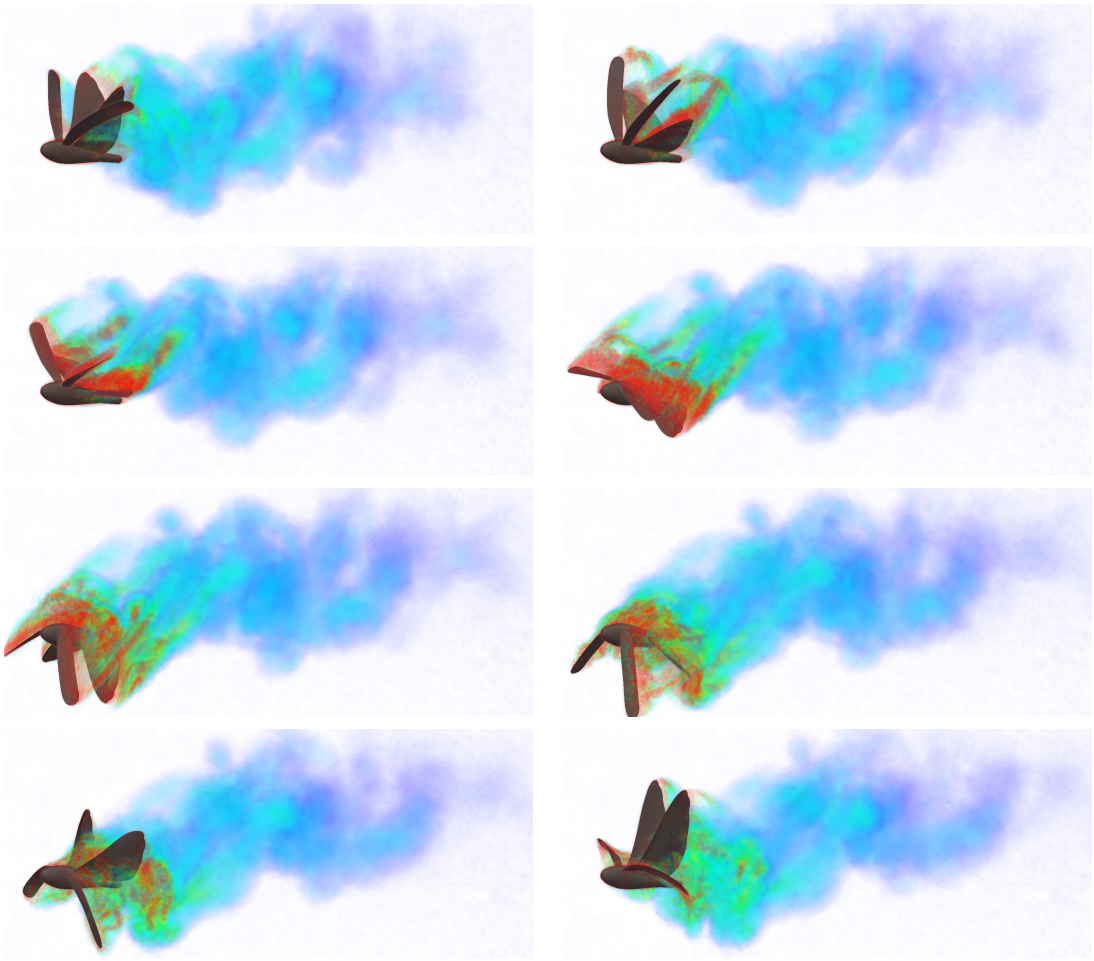


Figure 5.9: Volume rendering of the vorticity magnitude for the eight equally-spaced points during the third flapping cycle (left to right and then top to bottom).

Chapter 6

Test Computation with Increased Temporal Resolution

In this chapter we present from [11] various test cases for evaluating the solution accuracy based on increasing the resolution in time. Here, there are two types of studies: one changes the basis function orders used to represent the motion, which results in slightly different motion, and the other does not change the motion but instead changes the temporal discretization. We accomplish the temporal discretization in two different ways. In the first way, we increase the number of space–time slabs. In the second way, we increase the number of remeshes per flap cycle. This allows us to reduce the element distortion during mesh motion and verify that the element quality degradation of the base computation does not affect the results.

6.1 Temporal Setup

Changing the order of the basis functions in time and increasing the temporal discretization require very different techniques and have a different impact on the accuracy of the results.

6.1.1 Temporal Order

In this case, we change the order of the basis functions used to represent the wing motion. As described in [9], the wing motion in the base computation is represented using a third-order NURBS curve to obtain continuous acceleration. To show the affects of using lower-order NURBS curves, we reconstruct the wing motion using quadratic basis functions. To do this, we go back to the "tracking points" and use quadratic NURBS basis functions to represent the motion as described in [9]. The rest of the process is identical. As shown in Figure 6.1, the tip trajectories of the two different orders are not identical but very close.

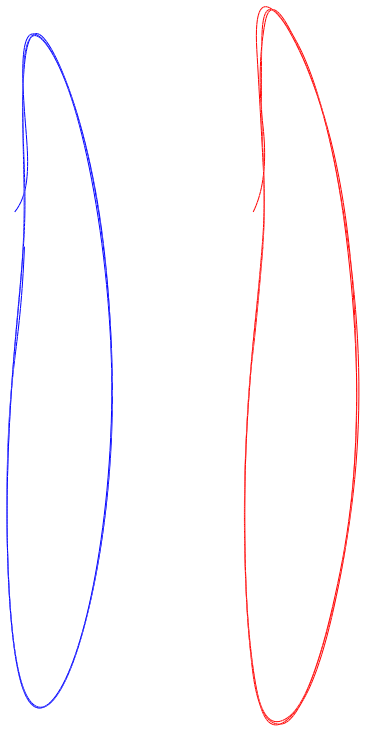


Figure 6.1: Temporal interpolations: cubic on left (blue) and quadratic on right (red).

6.1.2 Temporal Subdivision

As described in Section 5.3, we use a certain number of space–time slabs per knot span in the temporal representation of the mesh. We increase the number of slabs from 25 in the base case to 50 and 100 to analyze the effects (see Figure 6.2).

6.1.3 Remeshing

We use knot insertion to divide the flap cycle into four patches, where the distances traveled by the HW are approximately equal. Although four remeshes were enough to keep the mesh from tangling during mesh motion, we want to test whether the mesh quality resulting from having only four remeshes would affect the solution. We take each of the four temporal patches and insert knots once more to create eight temporal patches. Again, in each temporal patch, the distance traveled by the HW is approximately the same (see Figure 6.3).

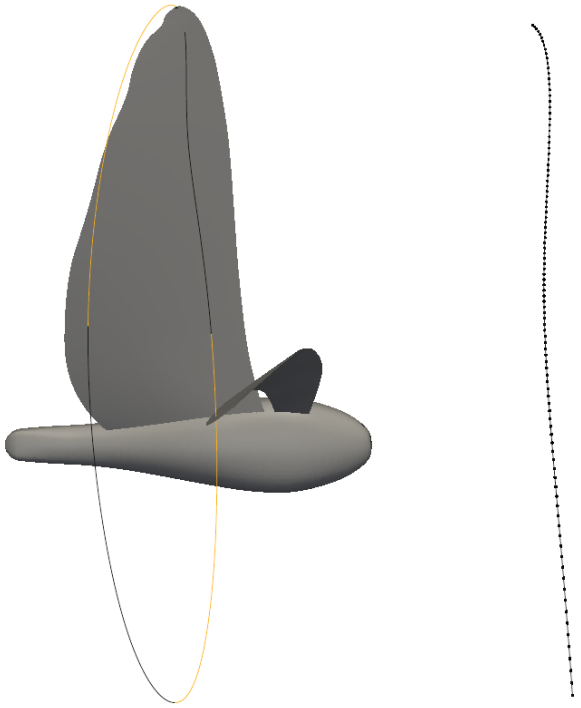
6.2 Temporal-Refinement Studies

6.2.1 Temporal-Order Study

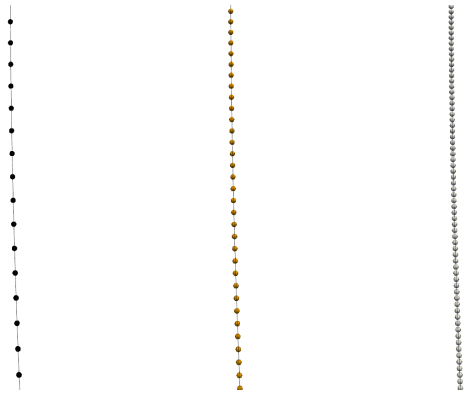
The total lift and thrust time histories from the cases in Section 6.1.1 can be seen in Figure 6.4. The discontinuities in the acceleration are clear when viewing the plot for the quadratic interpolation.

6.2.2 Temporal-Subdivision Study

The time histories of the total lift and thrust from the cases in Section 6.1.2 can be seen in Figure 6.5. There is little deviation between the three results. Although higher subdivision cases show a small increase in the maximum lift obtained. The overall percentage difference in average lift between the 25 and 50 slab cases is 2.6%,



(a) Full flap cycle divided into four patches (left) and magnified first patch (right).



(b) Single patch with 25 (left), 50 (middle), and 100 (right) space–time slabs per knot span.

Figure 6.2: Temporal discretization for the three different subdivisions.

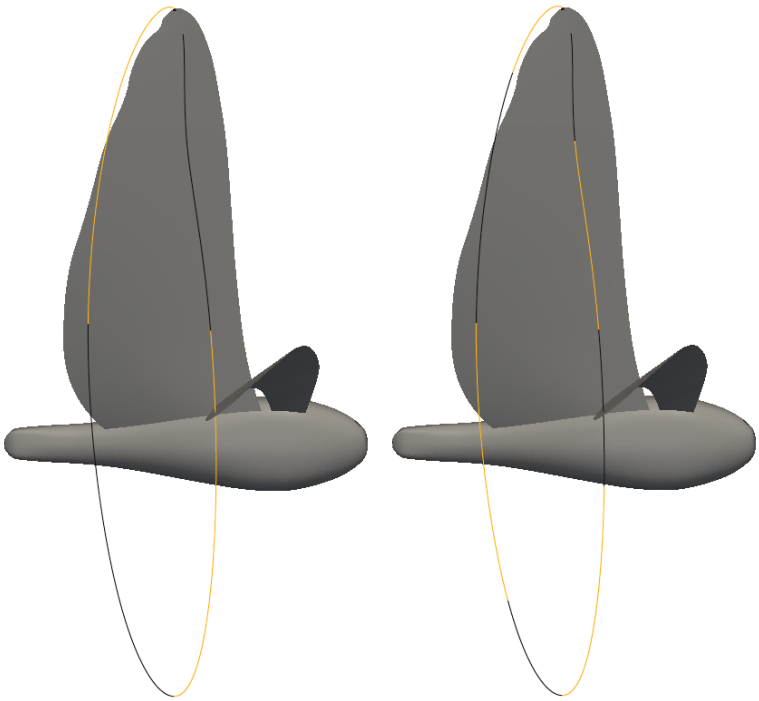


Figure 6.3: Base temporal split (left) and double temporal split (right).

and between the 25 and 100 slab cases is 6.4%. The difference in average thrust is also small at 4.2% and 4.8%, which corresponds to an average difference of only 0.06 mN.

6.2.3 Remeshing Study

The total lift and thrust time histories from the cases in Section 6.1.3 can be seen in Figure 6.6. There is hardly any difference between the two results. The percentage difference in average lift is 0.2%, showing that the base remesh rate is sufficient.

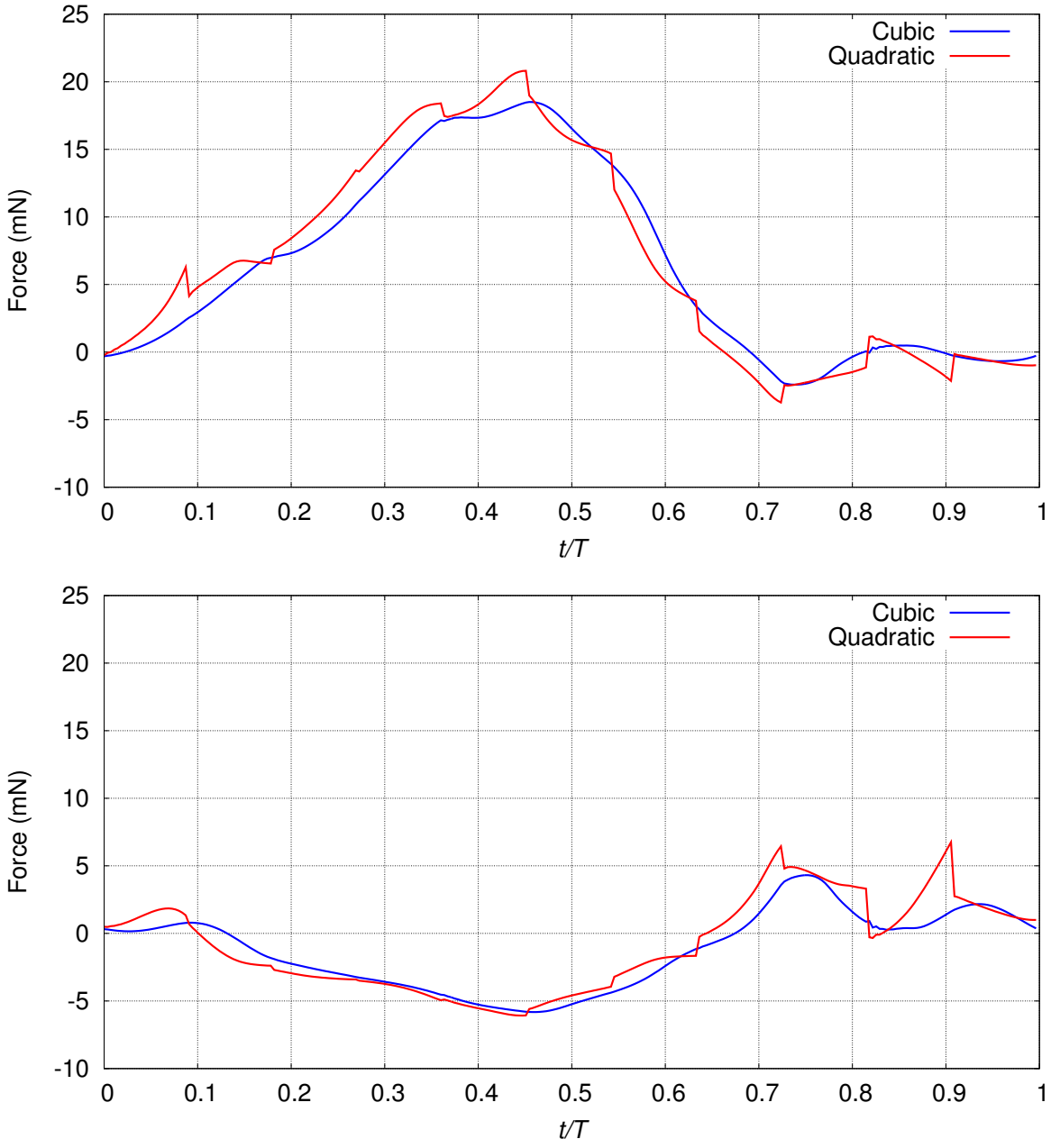


Figure 6.4: Temporal-order study. Total lift (top) and thrust (bottom).

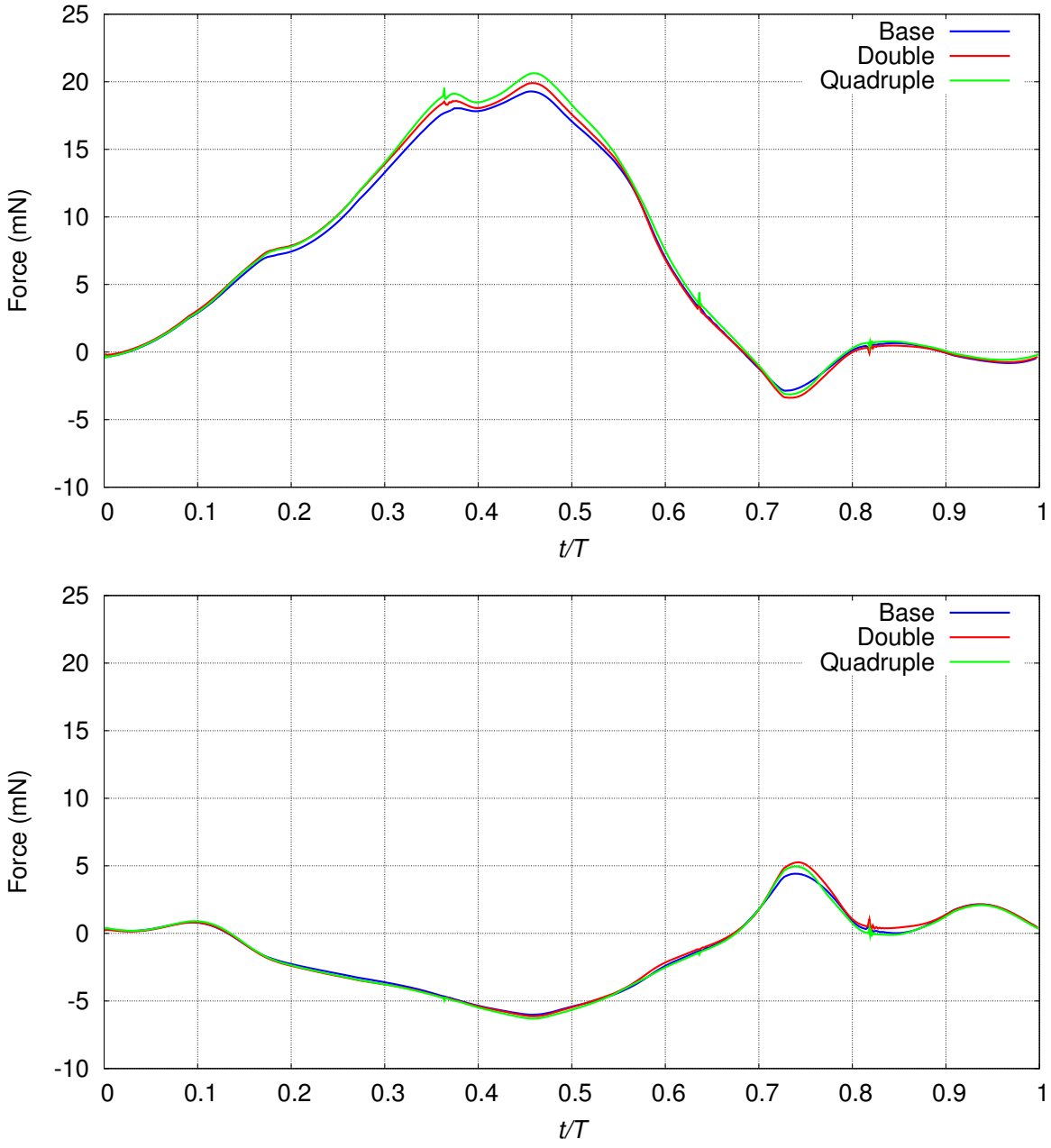


Figure 6.5: Temporal-subdivision study. Total lift (top) and thrust (bottom).

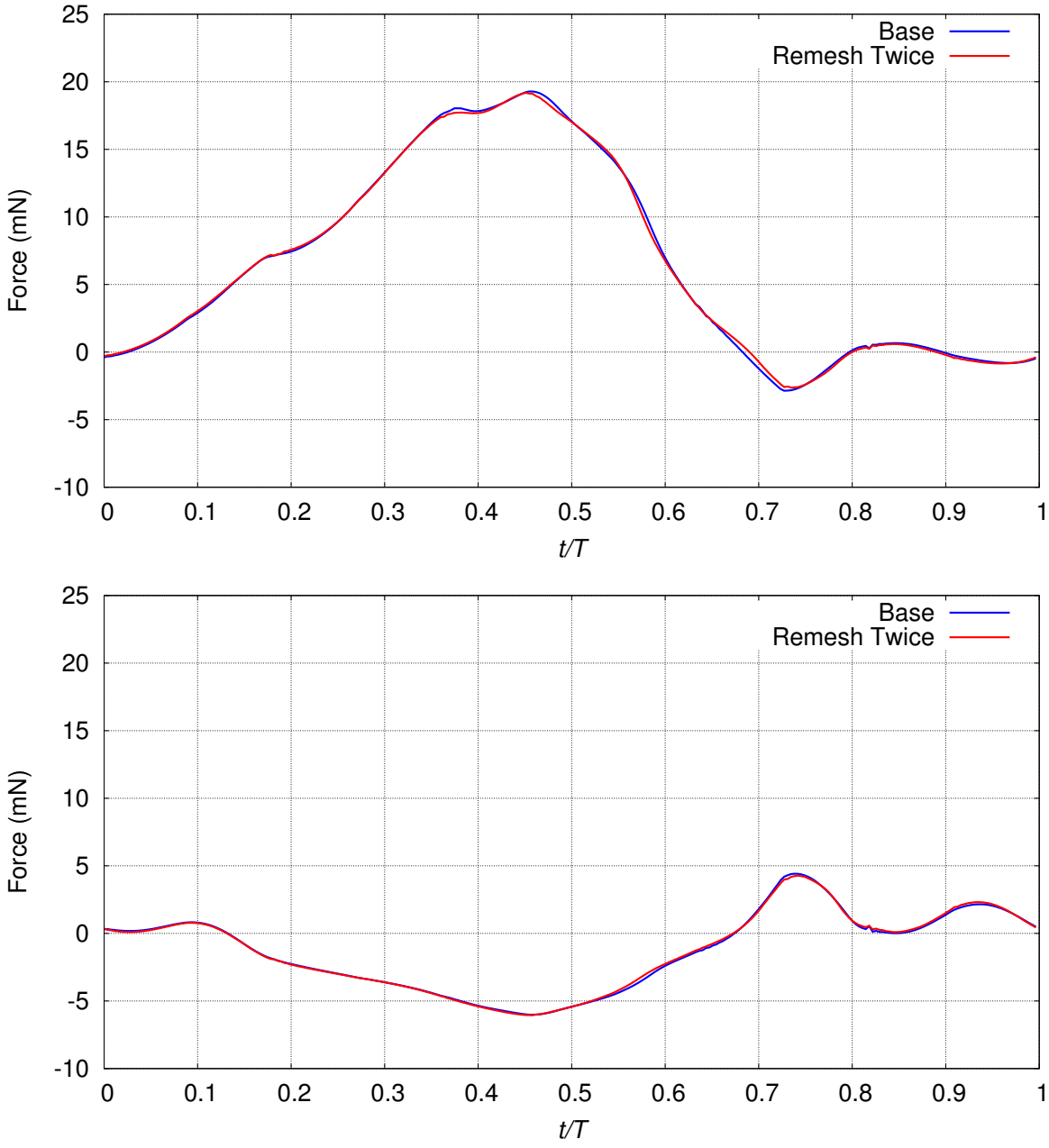


Figure 6.6: Remeshing study. Total lift (top) and thrust (bottom).

Chapter 7

Test Computations with Increased Spatial Resolution

To verify the accuracy of the results in terms on their dependency on the mesh resolution, we present from [11] a series of tests by refining the mesh. The main areas of interest include the wing-mesh refinement in the normal and tangential directions. In addition, the large number of node-to-node connections at the wingtips due to degeneration is addressed.

7.1 Refined-Mesh Generation

In this section the refinement of the finite element mesh based on the NURBS wings will be explained. All the mesh motion and time resolution parameters remain the same as in the base case, except where explicitly noted.

7.1.1 Refinement in the Normal Direction

For refinement in the normal direction, we increase the number of element layers in the refinement region normal to the wing surface from 1 to 10. The thickness of the

region with refined layers of elements also increases from the base 10% of the FW root chord to approximately 24% of the FW root chord (see Figure 7.1). The refined-mesh case requires more frequent remeshing due to the different aspect ratio of the volume elements. Because of this, we split the first temporal patch into two patches to increase remeshing frequency. As mentioned in Section 6.2.3, this, however, has a negligible effect on the solution.

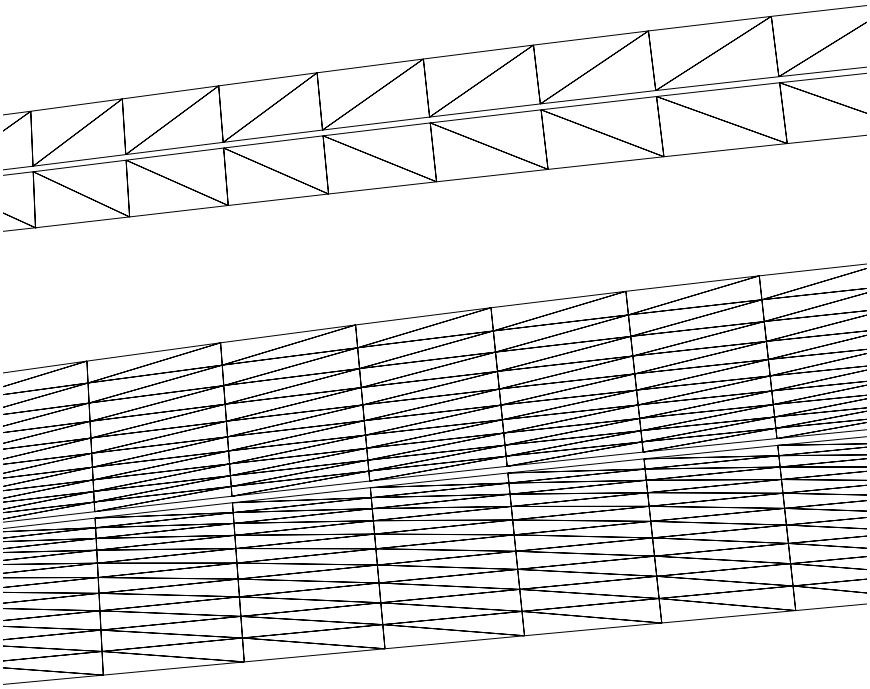


Figure 7.1: Base (top) and high-resolution (bottom) mesh refinement in the normal direction.

7.1.2 Refinement in the Tangential Direction

For refinement in the tangential direction, the mesh refinement is increased, approximately, by a factor of two in each of the wing surface parametric directions. Because the wing surfaces are represented by NURBS surfaces, generating a new finite element mesh with greater tangential refinement can be done with little effort. Refinement of the mesh in the tangential direction also creates a need for refinement of the body

mesh. Because of this, the body mesh is also refined, approximately, by a factor of two (see Figure 7.2). As in Section 7.1.1, the refined mesh requires more frequent remeshing, and, therefore, the first temporal patch is split into two patches. In addition to the refinement in the tangential direction, we add a bit more refinement in the normal direction compared to the base case. The thickness of this normal-refinement region is also 24% of the FW root chord as in Section 7.1.1, but with 4 element layers instead of 10.

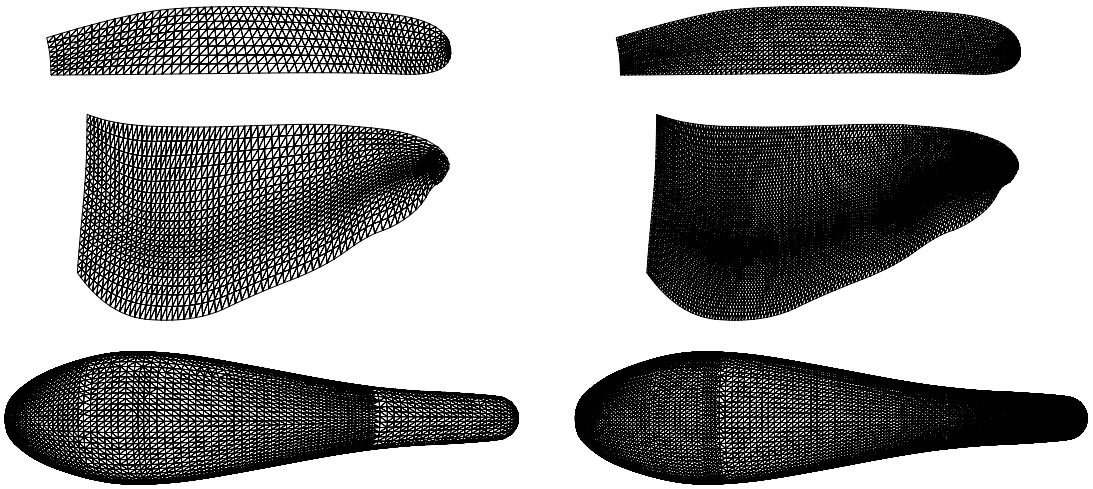
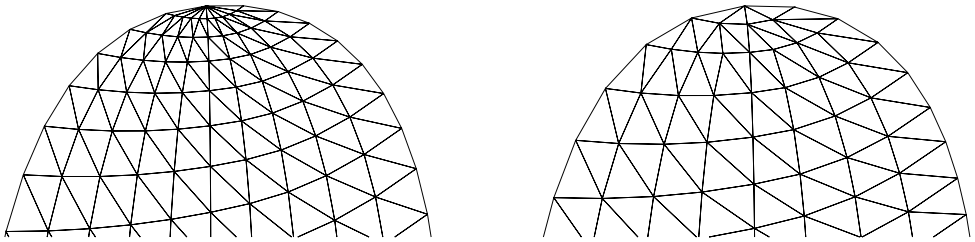


Figure 7.2: Surface meshes with base (top) and high-resolution (bottom) refinement in the tangential direction.

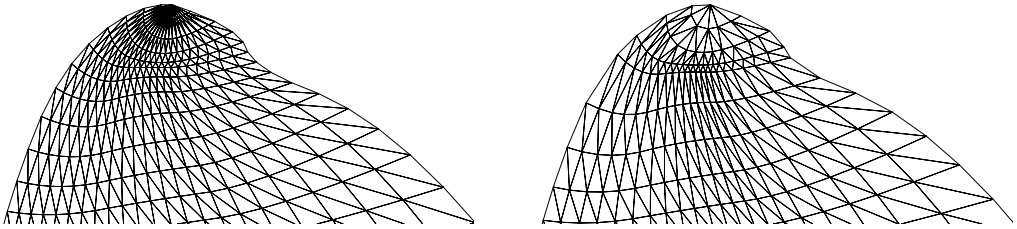
7.1.3 Patch Degeneration at Wingtips

We generate a finite element surface mesh for the wing by interpolating the NURBS geometry at each knot intersection and creating quadrilateral elements. We then subdivide each quadrilateral element into two triangles. Because an entire edge of a NURBS patch surface is degenerated into a single control point, the finite element mesh has a high number of node-to-node connections at the wingtips. To see if this has an effect on the results, we also generate a modified mesh using a NURBS-based design software where we reduce the number of elements at the wingtip (see

Figure 7.3).



(a) Base (left) and modified (right) FW tip mesh.



(b) Base (left) and modified (right) HW tip mesh.

Figure 7.3: Mesh comparisons around wingtips.

7.2 Studies on Spatial Mesh Refinement

7.2.1 Refinement in the Normal and Tangential Directions

The total lift and thrust time histories from the computation of the cases in Sections 7.1.1 and 7.1.2 can be seen in Figure 7.4. For most of the flap cycle there is little to no deviation between the base case and the two refinement cases. At the maximum and minimum of the two force curves there are some deviations, but the overall average force for a flap cycle changes little between the cases. The percentage difference between the average lift is 2.6% for the normal-refinement versus base case and 2.4% for the tangential-refinement versus base case. The percentage difference for the average thrust is 1.7% for the normal-refinement versus base case and 1.8%

for the tangential-refinement versus base case.

7.2.2 Patch Degeneration at Wingtips

The total lift and thrust time histories from the computation of the cases in Section 7.1.3 can be seen in Figure 7.5. For most of the flap cycle there is little to no deviation between the base and modified refinements. The base case yields a slightly higher and lower maximum and minimum lift, yet the percentage difference between the two is only 1.9%.

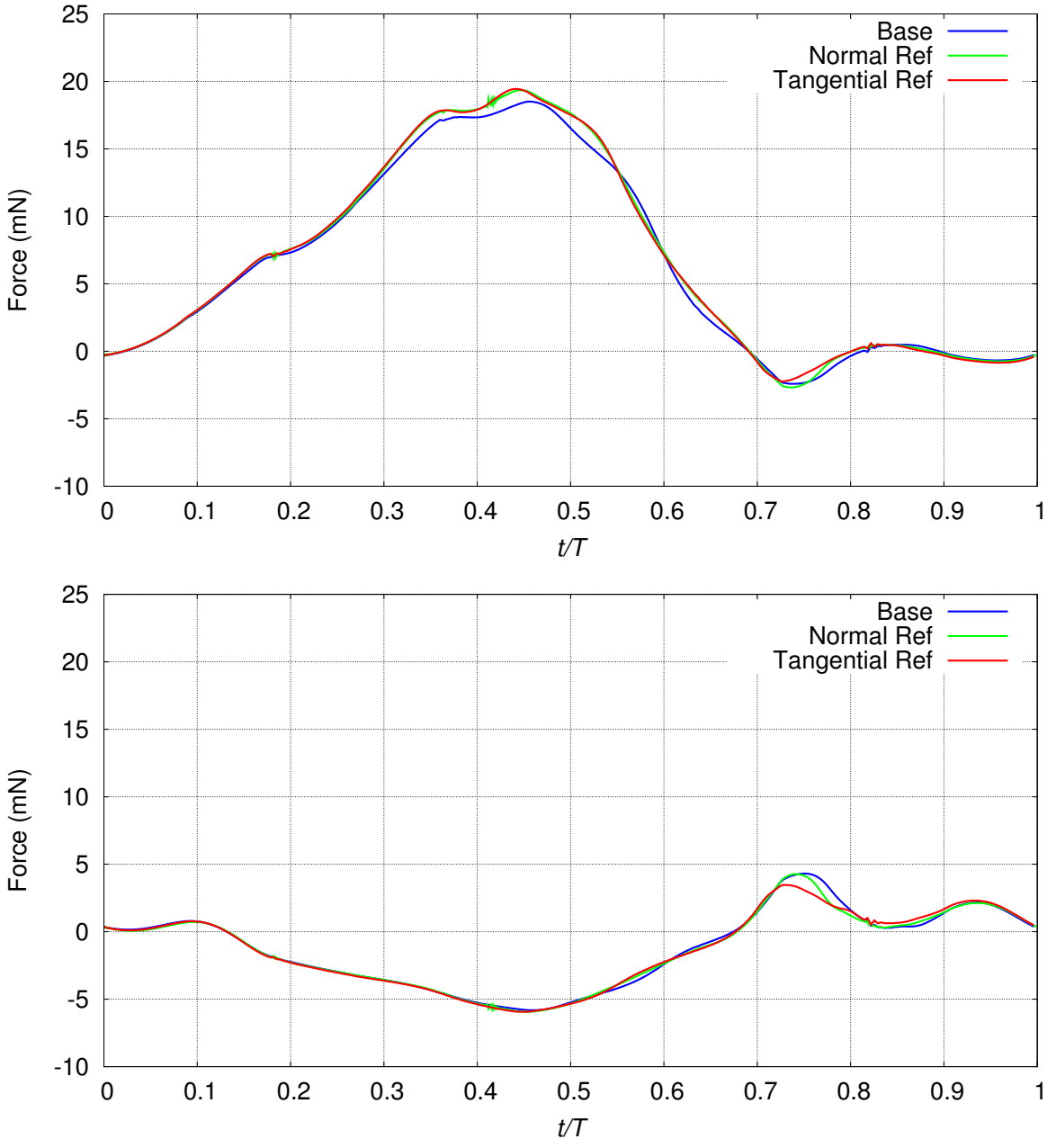


Figure 7.4: Studies on mesh refinement in the normal and tangential directions. Total lift (top) and thrust (bottom).

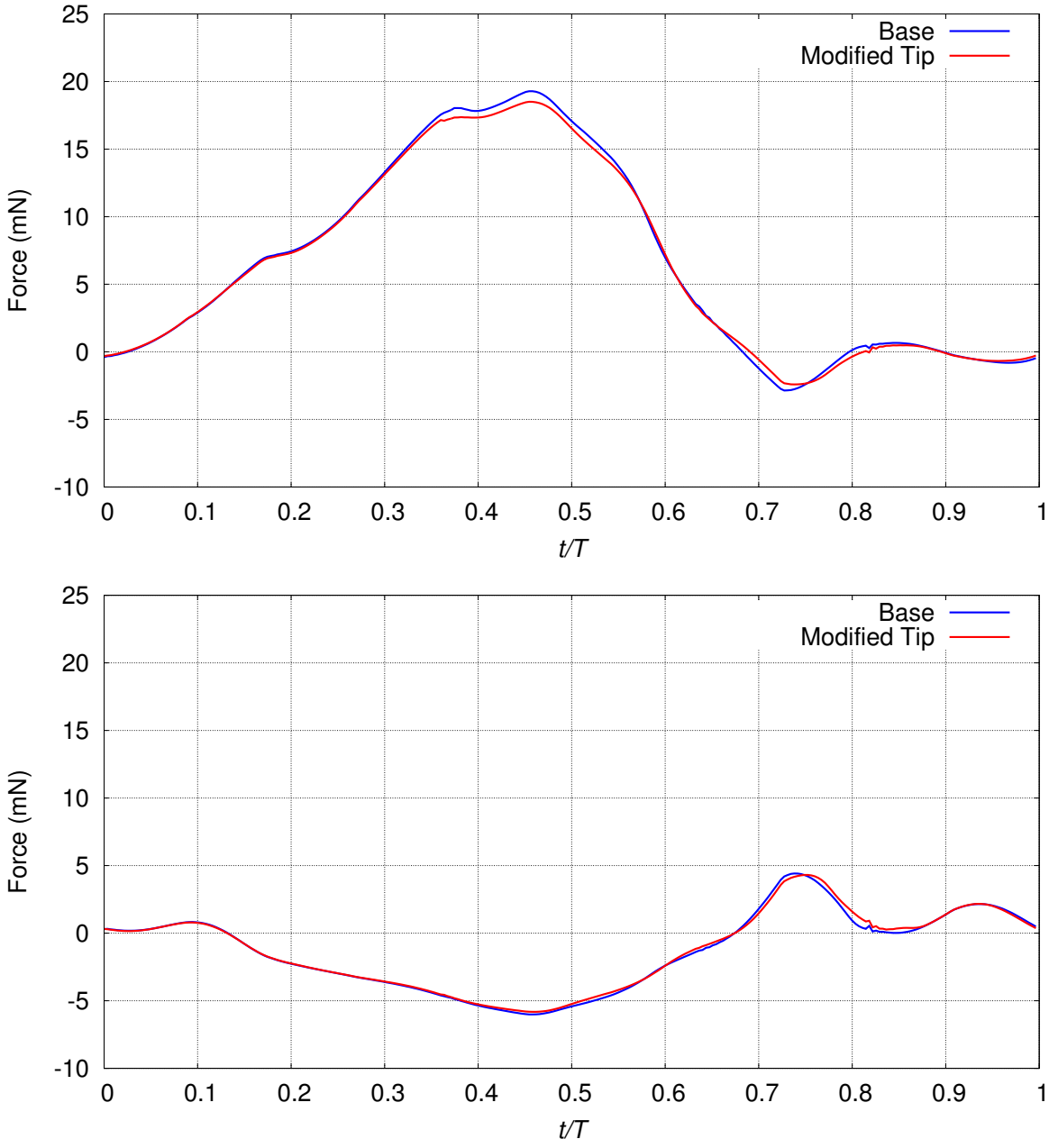


Figure 7.5: Studies on patch degeneration at wingtips. Total lift (top) and thrust (bottom).

Chapter 8

Wing Configuration Studies

This chapter, reporting material from [11], is a joint work with Nikolay Kostov. Once a sufficient level of refinement that produces consistent results has been found, we perform several tests that can be used for the future design of flapping wings of an MAV. We analyze the interaction between the FW and HW by creating a test case with the FW flapping and the HW removed, and a case with the HW flapping and the FW removed, named "FW Only" and "HW Only," respectively. This allows us to study the presence of any beneficial or disruptive interactions. In addition, the interaction between the left and right sets of wings is of interest. To see if there is a beneficial interaction between the left and right wings, we compute a case where only the wings on the right are present, named "Right," and a case where the wings on both sides are present, but the left wings are in a fixed position in the middle of the downstroke while the wings on the opposite side flap as normal. This case is named "Right and Fixed." The latter case allows for a pressure build up on the fixed wing, but minimizes interaction between the wings at the top of the stroke. Lastly, we want to see what the effect of wing camber and twist are, so we compute a test case where the wing is modified to a flat plate; this case is called "Flat Wings." A summary of the cases is provided in Table 8.1.

Case	Description
FW Only	Flapping FW with HW removed
HW Only	Flapping HW with FW removed
Right	Flapping right wings with left wings removed
Right and Fixed	Flapping right wings with left wings fixed
Flat Wings	Wings with camber and twist removed

Table 8.1: Summary of cases analyzed.

8.1 Setup

8.1.1 FW Only and HW Only

For the FW Only and HW Only cases the setup is very similar to the base case, other than the absence of the removed wings. The prescribed wing flapping motion is the same as for the base case. The mesh motion within the refinement cylinder is generated in the same way. The number of nonlinear iterations, GMRES iterations, and time-step size are kept the same as well.

8.1.2 Right, and Right and Fixed

For the case with wings only on the right side, we generate a mesh which does not have the left side wings. Everything else stays the same as the base case. For the "Right and Fixed" case, we generate the mesh in the same way but with the left wings stationary in the middle of the downstroke.

8.1.3 Flat Wings

Lastly, to generate the Flat Wings case we project all points for the wing onto the plane defined by the wingtip point, the leading edge point on the body, and the trailing edge point on the body. This way we remove the camber and twist of the deformed wing while keeping the same wingtip motion. The MAV body with flat wings is shown in Figure 8.1.

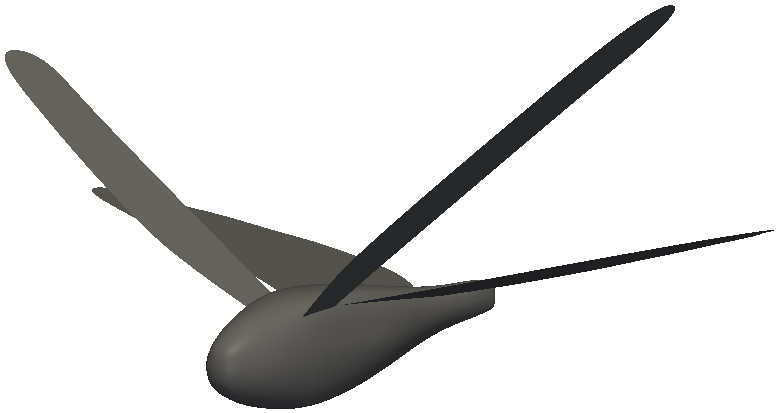
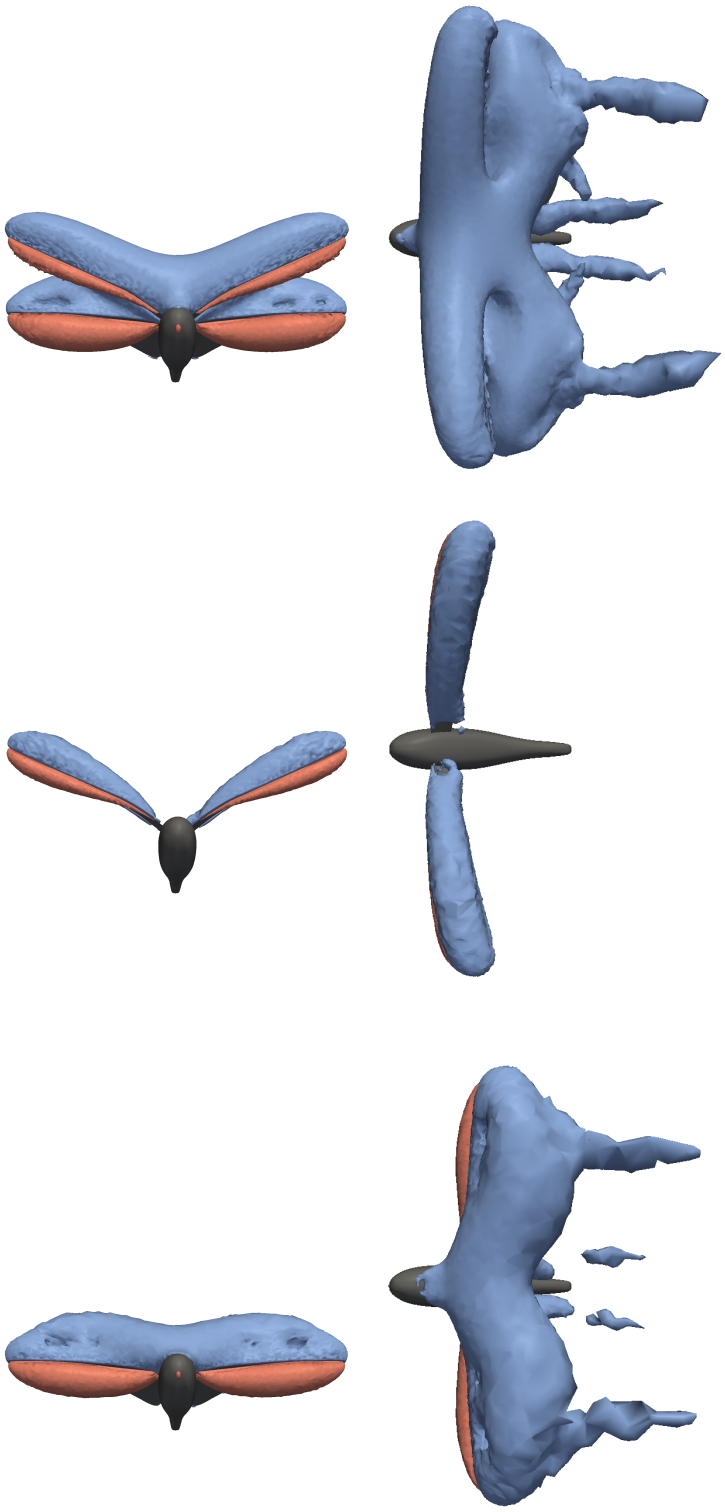


Figure 8.1: MAV with flat wings.

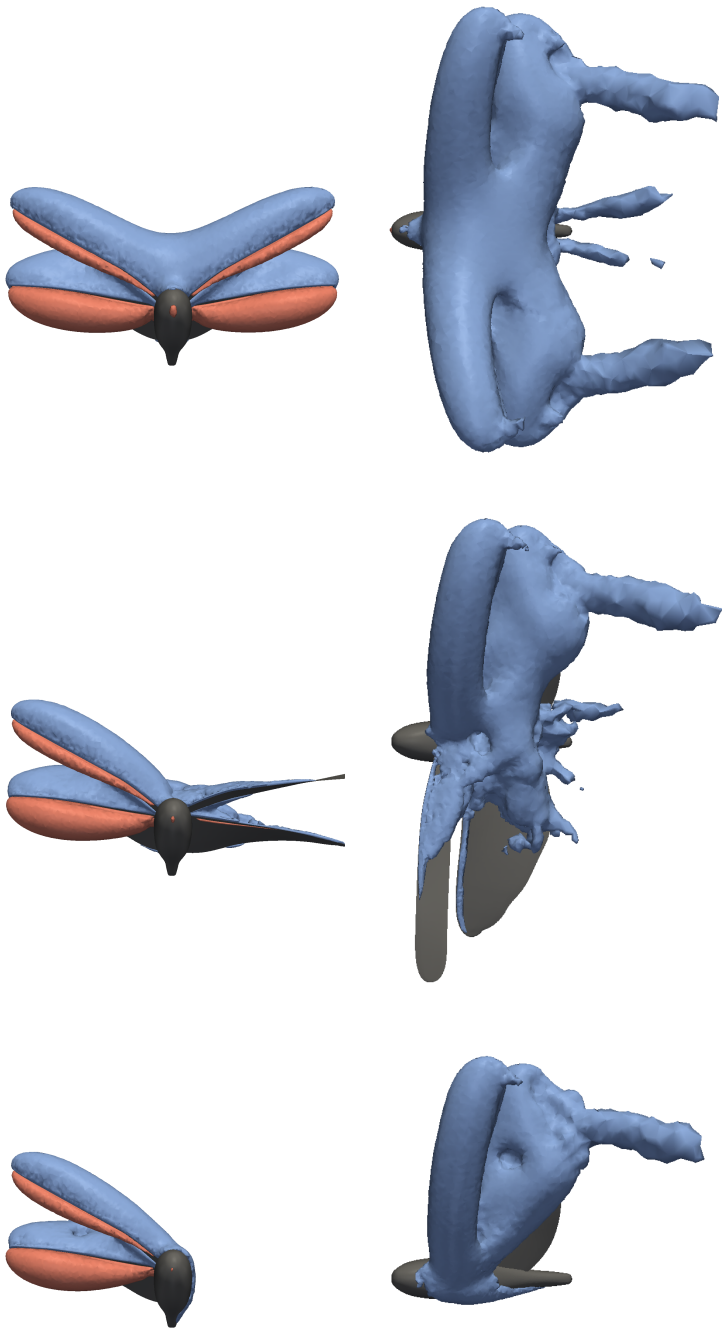
8.2 Results and Discussion

We compare the lift and thrust generated by the right wings in the FW Only, HW Only, Right, Right and Fixed, and base cases to evaluate the effect from the wing interactions on the force magnitudes. The isosurfaces of pressure around the FW and HW for the FW Only, HW Only, and base cases are shown in Figure 8.2, and for the Right, Right and Fixed, and base cases in Figure 8.3.



(a) Front view. (b) Top view.

Figure 8.2: Isosurfaces of pressure for base (top), FW Only (middle), and HW Only (bottom) cases.



(a) Front view. (b) Top view.

Figure 8.3: Isosurfaces of pressure for base (top), Right and Fixed (middle), and Right (bottom) cases.

8.2.1 HW Only and FW Only

The results for FW Only and HW Only cases are presented in Figures 8.4 and 8.5, respectively. Jensen [7] studied the aerodynamics of locust flight in great detail and discussed how during the downstroke the circulation from the FW decreases the effective angle of attack for the HW (thus decreasing lift on the HW) and the circulation from the HW increases the effective angle of attack for the FW (thus increasing lift on the FW). Our results are in good agreement with this observation as can be seen from the streamline plots shown in Figure 8.6. Using the streamlines to calculate the effective angle of attack for the FW, the increase is from 2.8° for the FW Only case to 12.5° for the base case. Overall, there is a net 9.2% increase in the lift generated by having two pairs of wings compared to the sum of the separate pairs, which corresponds to an average difference of 0.44 mN.

8.2.2 Right, and Right and Fixed

The results for the Right, and Right and Fixed cases are shown in Figures 8.7 and 8.8. For the case where only the right wings are present, as expected, there is a noticeable drop in the lift generated, especially in the middle of the downstroke. However, some of that loss of lift is due to the lack of a pressure build up on the part of the wing close to the body. It is evident that there is some beneficial interaction when there are flapping wings on both sides. There is a 12.5% net increase in average lift generation on the flapping side wings going from the Right and Fixed to the base case.

8.2.3 Flat Wings

Lastly, we analyze the results for the Flat Wings case. The comparison between the Flat Wings case and the base case is shown in Figure 8.9. We note that these forces are calculated by integrating over the entire wing and body surfaces. At the base

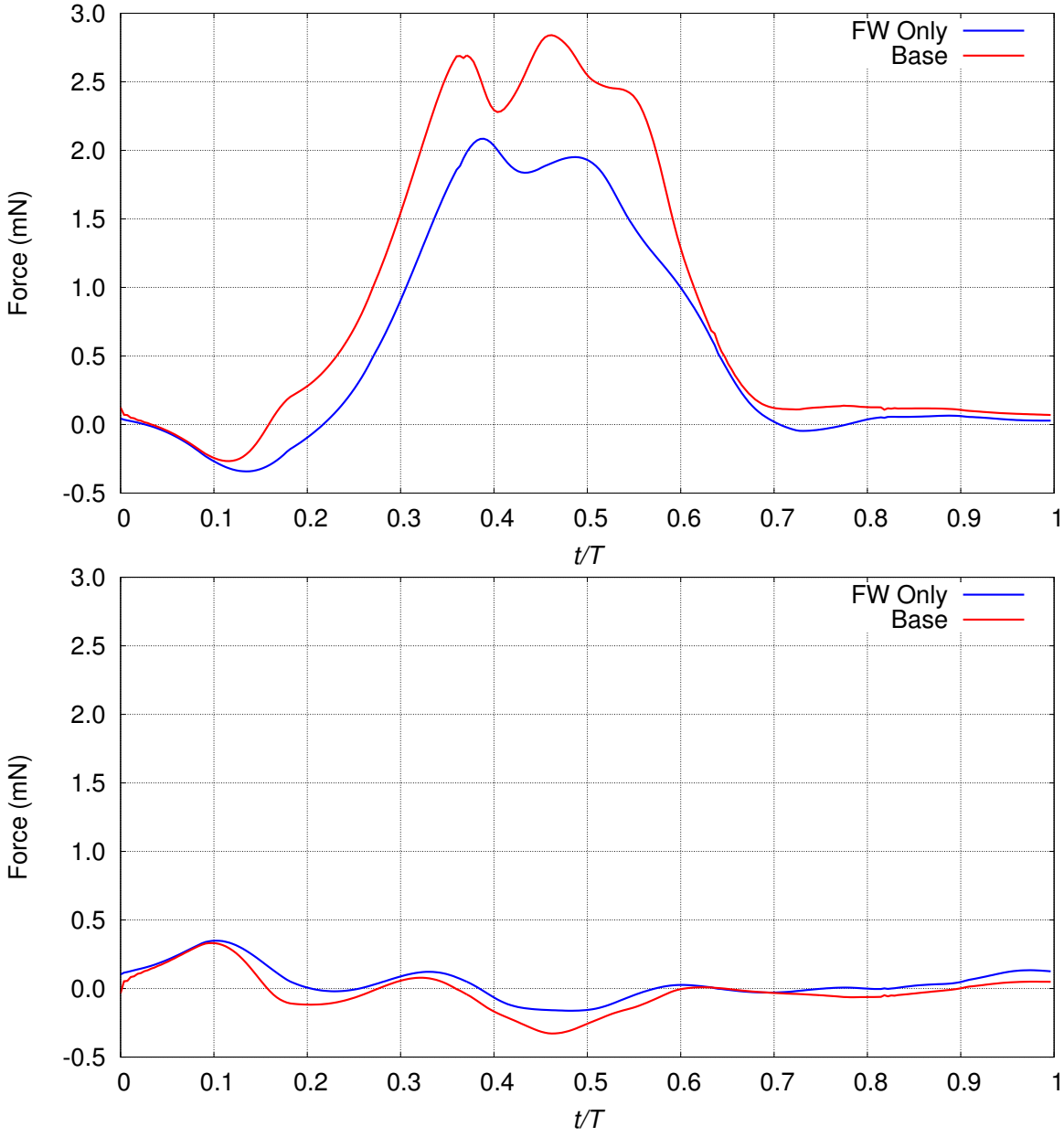


Figure 8.4: FW Only and base cases. Lift (top) and thrust (bottom) generated by the right FW.

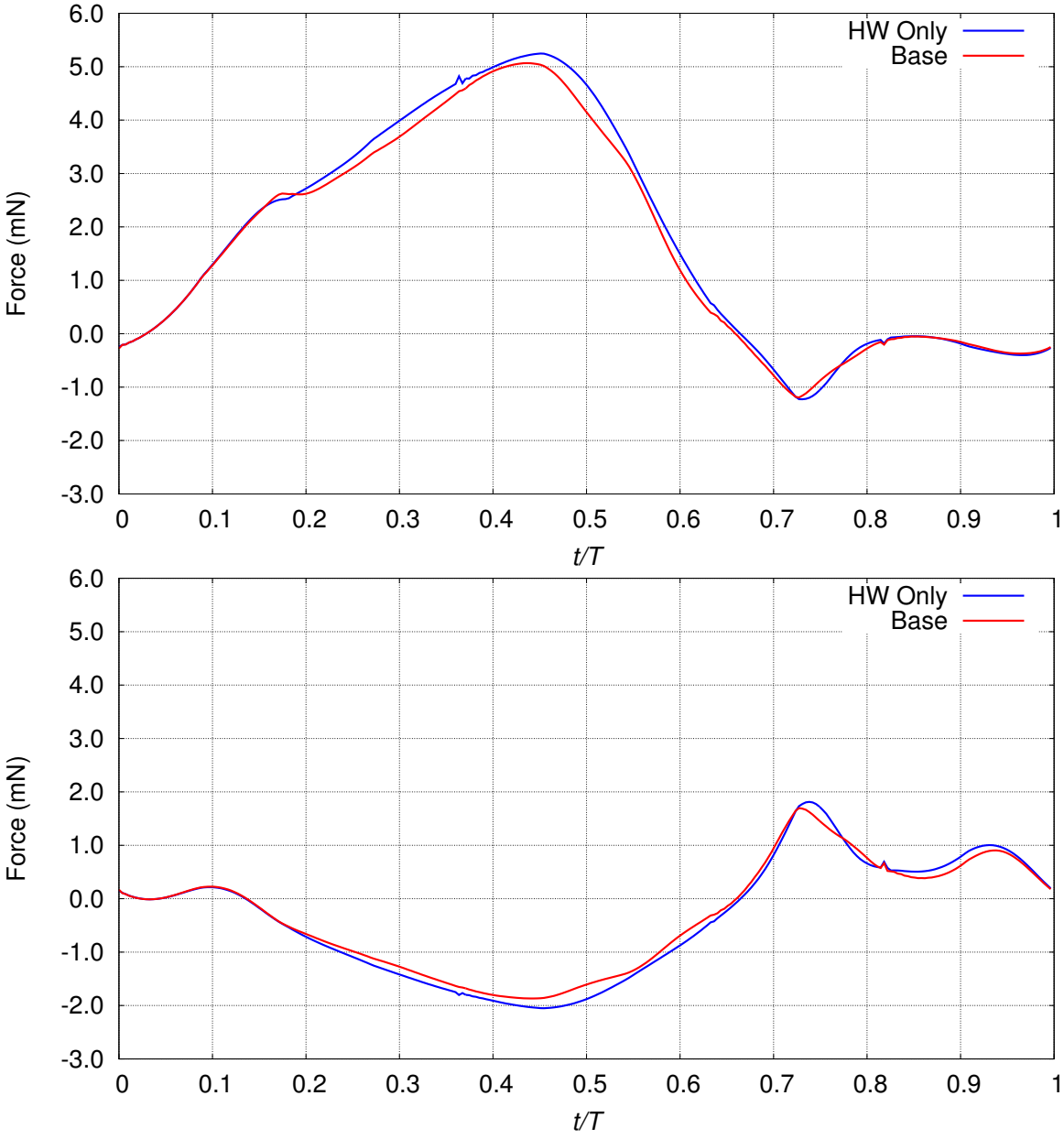


Figure 8.5: HW Only and base cases. Lift (top) and thrust (bottom) generated by the right HW.

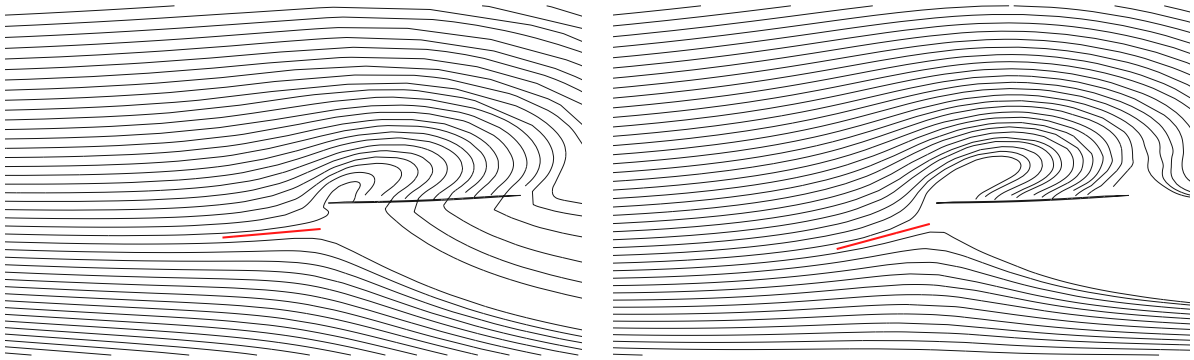


Figure 8.6: FW Only (left) and base (right) cases. Streamlines around FW vertical cross-section. Red line indicates the relative wind direction.

body angle the lift generated on the downstroke is similar between the Flat Wings case and the base case, but there is a large negative lift generated at the upstroke by the flat wing, which greatly reduces the average lift over a flapping cycle. In addition, the drag generated by the flat wings is much greater than the cambered and twisted wings, indicating that the forward velocity cannot be maintained with flat wings. The conclusion is that the main effect of wing camber and twist is to reduce drag and minimize the production of negative lift at the upstroke. Therefore, it is vital that a flapping-wing MAV utilizes wing camber and twist to maintain a high lift to drag ratio.

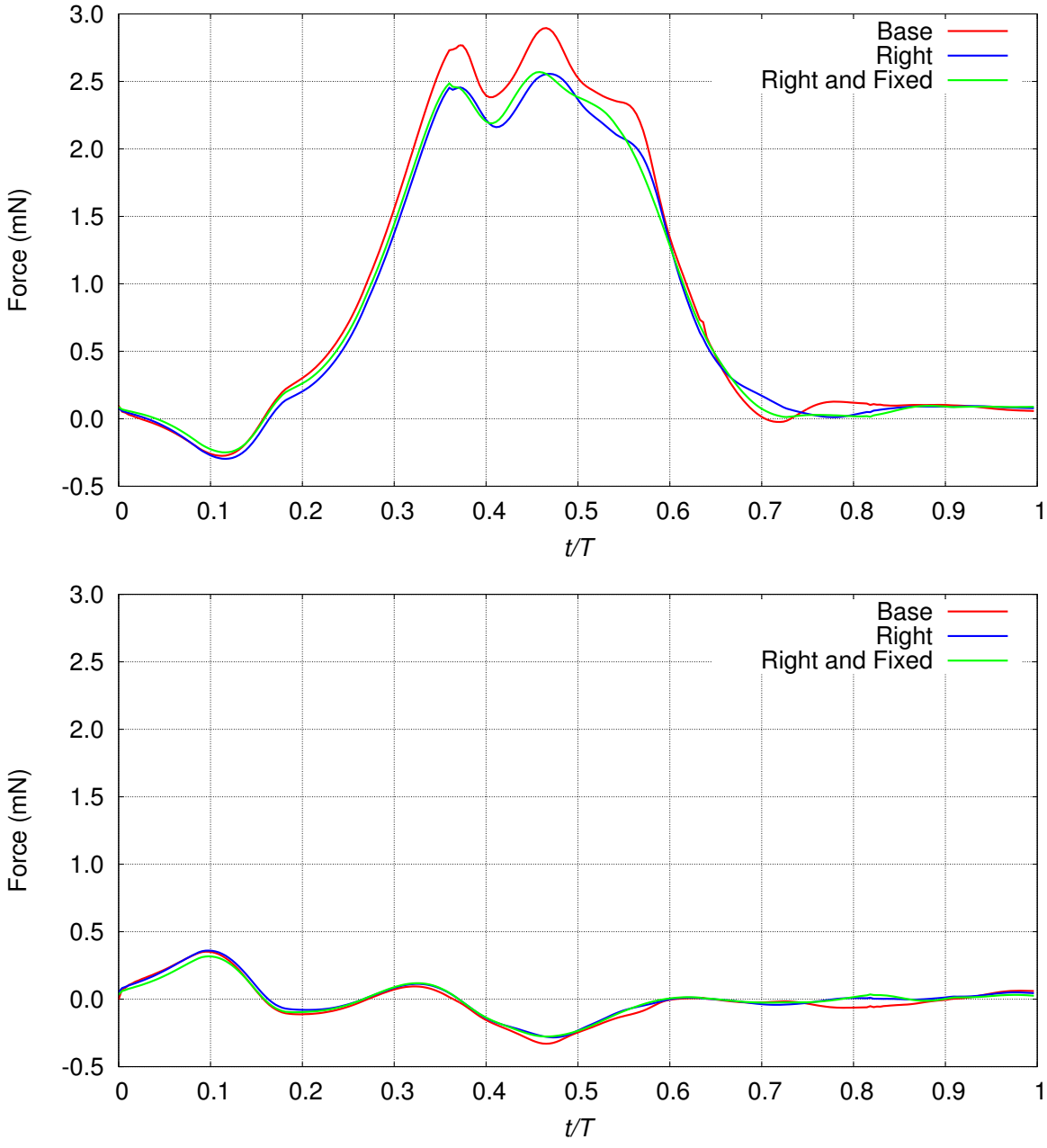


Figure 8.7: Base, Right, and Right and Fixed cases. Lift (top) and thrust (bottom) generated by the FW.

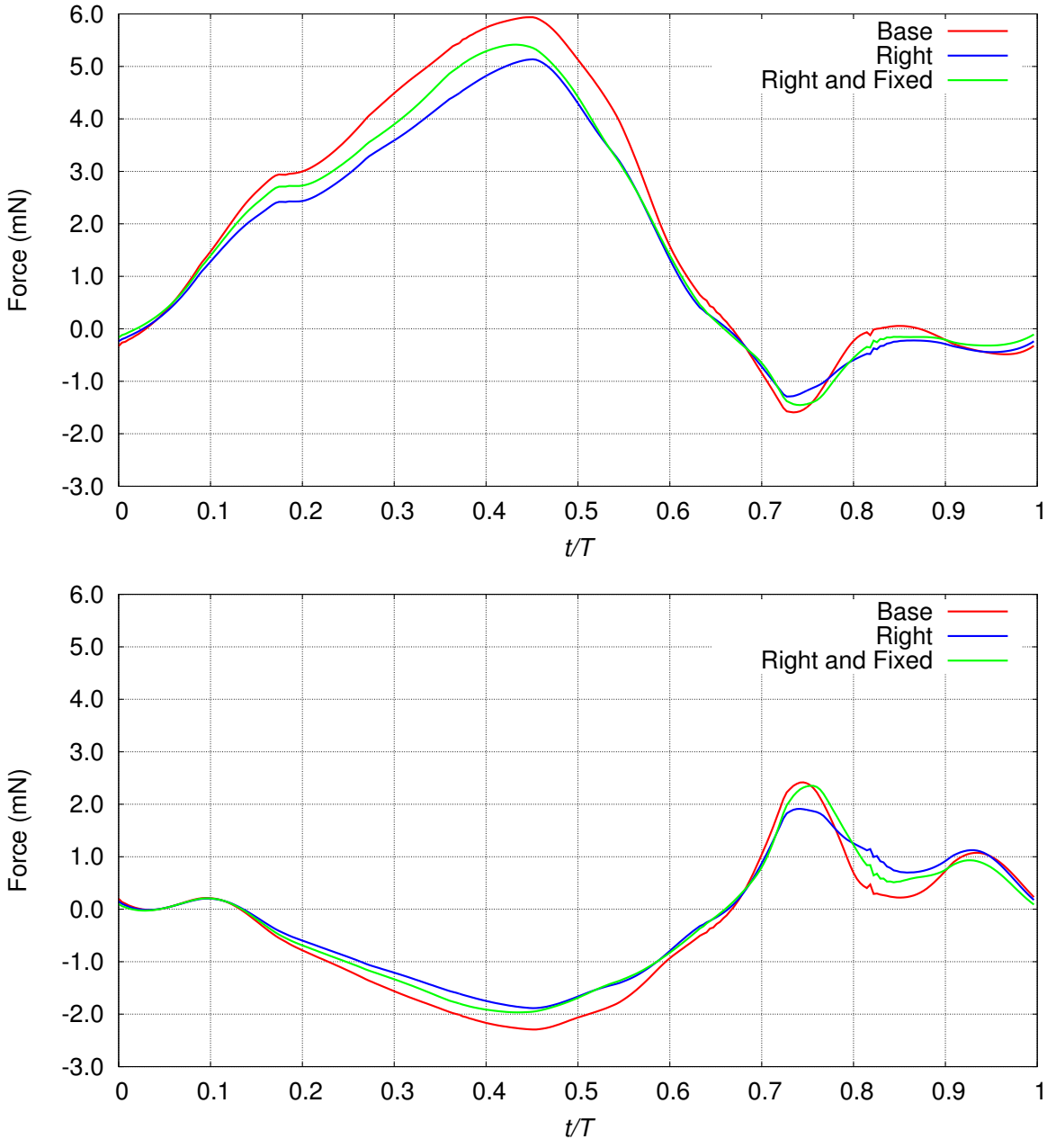


Figure 8.8: Base, Right, and Right and Fixed cases. Lift (top) and thrust (bottom) generated by the HW.

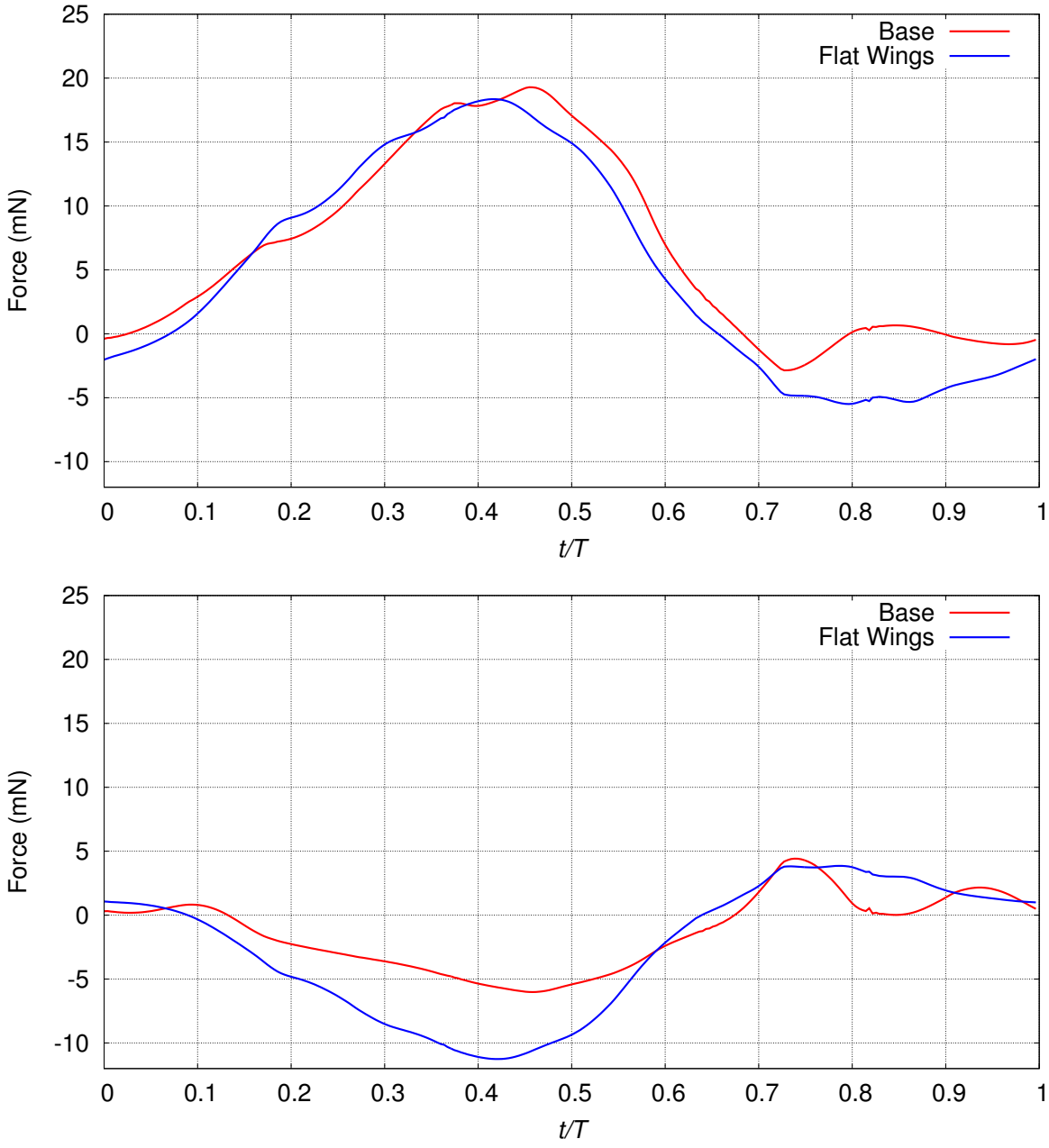


Figure 8.9: Flat Wings and base cases. Total lift (top) and thrust (bottom).

Chapter 9

Conclusions

Using the core and special space–time techniques developed by the T★AFSM recently, the focus of this thesis has been a detailed computational analysis of bio-inspired flapping-wing aerodynamics of an MAV. The computational techniques used include the DSD/SST formulation, specifically the version called "DSD/SST-VMST," which is the space–time version of the VMS method. The motion and deformation of the wings are based on data extracted from the high-speed, multi-camera video recordings of a locust in a wind tunnel. A set of special space–time techniques are also used in the computations in conjunction with the DSD/SST method. The special techniques are based on using, in the space–time flow computations, NURBS basis functions for the temporal representation of the motion and deformation of the wings and for the mesh moving and remeshing.

The computation of the base MAV case is presented first. After that spatial and temporal resolution studies are presented and compared to the base case. In terms of the temporal resolution, we separately tested increasing the temporal order, the number of temporal subdivisions, and the frequency of remeshing. In terms of the spatial resolution, we separately tested increasing the wing-mesh refinement in the normal and tangential directions and changing the way node connectivities are han-

dled at the wingtips. All of these computations show that the spatial and temporal resolutions used in the base case were sufficient. We also performed tests with different combinations of wing configurations for the MAV and made observations on the beneficial and disruptive interactions between the wings and the role of wing camber and twist.

Bibliography

[1] D. E. Alexander. *Nature's Flyers: Birds, Insects, and the Biomechanics of Flight*. The Johns Hopkins University Press, 2002.

[2] Y. Bazilevs and I. Akkerman. Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual–based variational multiscale method. *Journal of Computational Physics*, 229:3402–3414, 2010.

[3] Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197:173–201, 2007.

[4] M. H. D. et al. Wing rotation and the aerodynamics basis of insect flight. *Science*, 284:1954–1960, 1999.

[5] T. J. R. Hughes. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, 127:387–401, 1995.

[6] T. J. R. Hughes, A. A. Oberai, and L. Mazzei. Large eddy simulation of turbulent channel flows by the variational multiscale method. *Physics of Fluids*, 13:1784–1799, 2001.

[7] M. Jensen. Biology and physics of locust flight. III. The aerodynamics of locust flight. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 239:511–552, 1956.

[8] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal of Scientific Computing*, 20:359–392, 1998.

[9] K. Takizawa, B. Henicke, A. Puntel, N. Kostov, and T. E. Tezduyar. Space–time techniques for computational aerodynamics modeling of flapping wings of an actual locust. in preparation, 2012.

[10] K. Takizawa, B. Henicke, A. Puntel, T. Spielman, and T. E. Tezduyar. Space–time computational techniques for the aerodynamics of flapping wings. *Journal of Applied Mechanics*, 79:010903, 2012.

[11] K. Takizawa, N. Kostov, A. Puntel, B. Henicke, and T. E. Tezduyar. Space–time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle. in preparation, 2012.

[12] K. Takizawa and T. E. Tezduyar. Multiscale space–time fluid–structure interaction techniques. *Computational Mechanics*, 48:247–267, 2011.

[13] K. Takizawa and T. E. Tezduyar. Space–time fluid–structure interaction methods. *Mathematical Models and Methods in Applied Sciences*, to appear, 2012.

[14] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, and S. Mittal. Parallel finite-element computation of 3D flows. *Computer*, 26(10):27–36, 1993.

[15] T. E. Tezduyar. Stabilized finite element formulations for incompressible flow computations. *Advances in Applied Mechanics*, 28:1–44, 1992.

[16] T. E. Tezduyar. Computation of moving boundaries and interfaces and stabilization parameters. *International Journal for Numerical Methods in Fluids*, 43:555–575, 2003.

[17] T. E. Tezduyar, M. Behr, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: I. The concept and the preliminary numerical tests. *Computer Methods in Applied Mechanics and Engineering*, 94(3):339–351, 1992.

[18] T. E. Tezduyar, M. Behr, S. Mittal, and J. Liou. A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. *Computer Methods in Applied Mechanics and Engineering*, 94(3):353–371, 1992.

[19] T. E. Tezduyar and S. Sathe. Modeling of fluid–structure interactions with the space–time finite elements: Solution techniques. *International Journal for Numerical Methods in Fluids*, 54:855–900, 2007.

[20] S. M. Walker, A. L. R. Thomas, and G. K. Taylor. Deformable wing kinematics in the desert locust: how and why do camber, twist and topography vary through the stroke. *Journal of The Royal Society Interface*, 6:735–747, 2009.